REASSESSING THE BIOLOGICAL PLAUSIBILITY OF BACKPROPAGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Backpropagation is often called into question on the grounds of biological implausibility as well as its incompatibility with analog and neuromorphic hardware. Alternatives to backpropagation range from heuristic modifications to energy-based learning algorithms. However, the criteria for what constitutes biological plausibility vary and often a limited definition of backpropagation is used, which conflates the choice of algorithm and network topology, exclusively considering backpropagation in the feedforward/DAG setting. In this work, we revisit the method of Lagrange multipliers, which, when applied to credit assignment in a neural network, yields recurrent backpropagation with standard backpropagation as a special case. This frequently overlooked Lagrangian formulation of backpropagation, previously explored in LeCun (1988) possesses many of the merits and locality properties favored by the literature on local and bio-inspired learning. We highlight that recurrent backpropagation dynamics are fully local and that a common objection, the distinction between activity states and error states, can be bypassed with a simple change of coordinates, leading to what we call dyadic recurrent backpropagation. We demonstrate experimentally on networks with asymmetric topology that the gradient estimates obtained in both coordinate systems are equivalent, with parameter-matched models outperforming standard feedforward backpropagation in the asymmetric setting. We additionally derive a variant of dyadic recurrent backpropagation which may be more hardware friendly. These results suggest that backpropagation is substantially closer to biological plausibility than is typically assumed, and may be more compatible with analog and neuromorphic computing paradigms than previously believed as well.

1 INTRODUCTION

Recent advances in artificial intelligence are driven by two principal developments of the past 30 years: the widespread availability of commercial GPUs and the backpropagation (BP) of error algorithm (Rumelhart et al., 1986). The rapid growth of AI, however, has been accompanied by a host of growing concerns. In particular, the energy-cost of training and deploying this technology has increased substantially. As AI approaches human-level reasoning, accompanied by exploding energy costs, many have pointed out that biological reasoning is accomplished with extremely modest energy consumption by comparison, and that perhaps through addressing this fundamental discrepancy we might approach a more responsible path to machine intelligence. Some theoretical progress has been made in this direction, with estimates of energy-savings via local learning algorithms projected up to 4 orders of magnitude (Yi et al. (2023)), though implementation remains a significant challenge.

Previous efforts towards bridging the gap between biological and artificial learning have dismissed backpropagation as fundamentally inconsistent with biology. Firstly, where backpropagation is frequently assumed to be spatially non-local—synaptic updates being dependent on information transmitted from arbitrarily far downstream— biological synapses are assumed to utilize only information that is available locally in space. In this paper, we employ a change of coordinates to derive two local learning algorithms, *Dyadic RBP* and *Split Dyadic RBP*, which possess the same locality properties as allegedly bio-plausible algorithms including predictive coding, equilibrium propagation and contrastive Hebbian learning. Additionally, we clarify several misconceptions prevalent in the literature on biologically plausible learning algorithms, showing that RBP and BP alone can be viewed–with

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

a few caveats–as comparably "bio-plausible" to many previously proposed backprop-alternatives. Though our contribution is mainly theoretical, we also validate the algorithms in experiments employing asymmetrically connected networks.

1.1 BACKGROUND

The formulation of BP and recurrent backpropagation (RBP) as relaxation of a single objective composed of local terms, was to our knowledge first proposed by LeCun (1988). Despite this early seminal work, demonstrating RBP (and the special case of BP) as local learning algorithms, BP has often been portrayed as synonymous with a strictly ordered forward pass through the layers of a feedforward model followed by a backward pass. This misconception has motivated a number of biologically inspired learning backprop variants, such as contrastive Hebbian learning (Movellan, 1991; Xie & Seung, 2003), generalized recirculation (O'Reilly, 1996), equilibrium propagation (Scellier & Bengio, 2017), predictive coding (Whittington & Bogacz, 2017) and target propagation (Bengio, 2014; Lee et al., 2015).

In the literature on biologically plausible learning algorithms, the bio-plausibility of backpropagation has typically been characterized by several overarching criteria assumed *not* to be met by the brain (Crick, 1989; Whittington & Bogacz, 2019; Lillicrap et al., 2020). A comprehensive, but non-exhaustive list of these includes: *(i) weight transport* — error propagation requires feedback weights equal to the transpose of the forward weights; *(ii) a dedicated error channel* — errors and activity are distinct signals, signed and carried by a separate circuit; *(iii) non-local learning rules* — a synapse's update depends on errors computed arbitrarily far downstream rather than on locally available signals; and *(iv) update locking* — strict scheduling with buffered activations.

Most proposed backprop alternatives target a subset of these criteria. Weight transport (i) is addressed by fixed random feedback (Lillicrap et al., 2016; Nøkland, 2016), sign-symmetric feedback (Liao et al., 2016; Xiao et al., 2019), learned feedback (Kolen & Pollack, 1994; Akrout et al., 2019), target propagation (Bengio, 2014; Lee et al., 2015), and variants Meulemans et al. (2020; 2021). The error channel (ii) is eliminated in equilibrium propagation, where errors are perturbations of the same activity (Scellier & Bengio, 2017); dendritic models instead localize errors in apical compartments or burst multiplexing (Urbanczik & Senn, 2014; Guerguiev et al., 2017; Sacramento et al., 2018; Payeur et al., 2021); similarly, dual propagation treats errors as differences of oppositely perturbed states (Høier et al., 2023; Høier & Zach, 2024), where predictive coding instead embraces dedicated error neurons (Rao & Ballard, 1999). Locality (iii) a strength of predictive coding, whose updates are Hebbian products of local errors and activities (Rao & Ballard, 1999; Whittington & Bogacz, 2017; Millidge et al., 2021), as well as contrastive Hebbian learning schemes, whose weight updates are local activity contrasts (O'Reilly, 1996; Xie & Seung, 2003; Scellier & Bengio, 2017). Update locking (iv) is relaxed by direct feedback alignment and random target projection (Nøkland, 2016; Frenkel et al., 2021), as well as synthetic gradients (Jaderberg et al., 2017), continual equilibrium propagation(Ernoult et al., 2020), and incremental predictive coding (Salvatori et al., 2024). While most proposals address several at once, they tend to incur a significant cost in performance in terms of performance at scale (Bartunov et al., 2018).

In what follows we show that two variants of RBP expressed in dyadic coordinates can compute exact gradients under arbitrary connectivity, while avoiding distinct error-channels (ii), and being fully local (iii). We also partially resolve update locking (iv) by having errors and predictions propagate simultaneously although weights must be updated after both have converged. We therefore do not accrue the memory cost of storing buffered activations. Weight transport (i) remains partially unresolved: during inference, activity is transported by weights which may be asymmetric, but during training error signals propagate via the transpose of the weights. Standard techniques from the feedback alignment literature can of course be applied to make our algorithms avoid this issue entirely,at the cost of worse gradient estimates.

1.2 LEARNING WITH DYADIC NEURONS

Dyadic neurons, which encode predictions as averages and errors as differences, were introduced in the dual propagation algorithm for training of feedforward networks (Høier et al., 2023). A follow-up paper showed that variants of dual propagation (Høier & Zach, 2024) can be derived either via the optimal value reformulation (Outrata, 1988; Dempe & Zemkoho, 2013) or by applying a change

of coordinates to the Lagrangian associated with the credit assignment problem (LeCun, 1988), which is the approach we take in this work. Dual propagation is however limited to networks with feedforward connectivity and makes limiting assumptions regarding the exact form of the constraint function F. The dyadic learning algorithm (Høier et al., 2024) is a generalization of dual propagation to networks with feedback, derived via the optimal value formulation, but like dual propagation the constraints are assumed to be of a specific form. Recently Scurria et al. (2026), motivated by the Bateman–Galley formulation (Galley, 2013) of non-conservative systems, derived dyadic equilibrium propagation— a novel generalization of dyadic learning to a more general class of constraints F. Unlike dual propagation dyadic learning and dyadic equilibrium propagation implicitly assume that the mixing of dyadic neuron states is symmetric.

Our contribution here is a continuation of these works, combining insights from all of them. We combine the Lagrangian of RBP (LeCun, 1988) (rather than that of standard BP) with the dyadic reparameterization of (Høier & Zach, 2024) allowing asymmetric mixing of neuron states, while using general constraints as in (Scurria et al., 2026), yielding the two new algorithms *Dyadic RBP* and *Split Dyadic RBP*. Additionally we demonstrate the applicability of the derived algorithms for training asymmetric convolutional networks for both supervised classification and unsupervised denoising, whereas prior works in the dyadic learning literature has been limited to shallow asymmetric MLPs and CNNs with feedforward connectivity.

2 The Lagrangian Formulation of Recurrent Backpropagation

Backpropagation is usually derived as the repeated application of the chain rule of calculus to a composed function. This gives rise to the forward pass (evaluating the function) and backward pass (applying the chain rule to compute derivatives with respect to hidden layers and weights). LeCun (1988) instead derives backpropagation in feedforward networks as well as recurrent backprop in recurrently connected networks using the method of Lagrange multipliers. In this section we briefly reiterate this derivation in a manner that makes minimal assumptions on the exact form of the constraints.

The neural network training problem can be stated as the following optimization problem

$$\min_{\theta} C(s) \quad \text{s.t.} \quad F(\theta, s) = 0, \tag{1}$$

where $F(\theta, s) = 0$ enforces the desired (typically nonlinear) behavior of the neurons[1] and θ denotes the set of trainable parameters. We define the Lagrangian $\mathcal{L}(\theta, s, \delta)$, which joins the cost and the constraints in a single objective.

$$\mathcal{L}(\theta, s, \delta) = C(s) + \delta^\top F(\theta, s) \tag{2}$$

For consistency with standard neural network notation we use δ to denote the Lagrange multipliers rather than λ which is usually used in the optimization literature. The gradient of the cost C with respect to θ can be computed when s and δ are at equilibrium — i.e. when

$$\frac{\partial \mathcal{L}(\theta, s, \delta)}{\partial s} = 0, \quad \frac{\partial \mathcal{L}(\theta, s, \delta)}{\partial \delta} = 0 \tag{3}$$

To reach equilibrium, recurrent backpropagation evolves s and δ so that each follows the negative gradient of $\mathcal{L}$ with respect to the other variable; the fixed points of these relaxation dynamics are precisely the stationarity conditions above.

$$\tau \dot{s} = -\frac{\partial \mathcal{L}(\theta, s, \delta)}{\partial \delta} = -F(\theta, s) \tag{4}$$

$$\tau \dot{\delta} = -\frac{\partial \mathcal{L}(\theta, s, \delta)}{\partial s} = -\frac{\partial C(s)}{\partial s} - \left(\frac{\partial F(\theta, s)}{\partial s}\right)^\top \delta := -G(s, \delta) \tag{5}$$

Here we have introduced the shorthand $G(s, \delta) = \frac{\partial C(s)}{\partial s} + \left(\frac{\partial F(\theta, s)}{\partial s}\right)^\top \delta$. Suppose the dynamics converge to an equilibrium (s^*, δ^*), and assume $\partial F/\partial s$ is invertible there, so that by the implicit

[1] Common choices include $s - f(Ws) = 0$, $f^{-1}(s) - Ws = 0$ and $s - Wf(s) = 0$. The two first have s represent *activity* whereas the latter has s represent *pre-activity*.

function theorem s^* and δ^* are well-defined and differentiable in a neighborhood of θ. By equation 4, $F(\theta, s^*(\theta)) = 0$ holds identically on this neighborhood, so $C(s^*(\theta)) = \mathcal{L}(\theta, s^*(\theta), \delta^*(\theta))$ as functions of θ, and their total derivatives coincide. Expanding the right-hand side by the chain rule,

$$\frac{dC}{d\theta} = \frac{\partial \mathcal{L}}{\partial \theta} + \underbrace{\frac{\partial \mathcal{L}}{\partial s}}_{=0} \frac{ds^*}{d\theta} + \underbrace{\frac{\partial \mathcal{L}}{\partial \delta}}_{=0} \frac{d\delta^*}{d\theta} \tag{6}$$

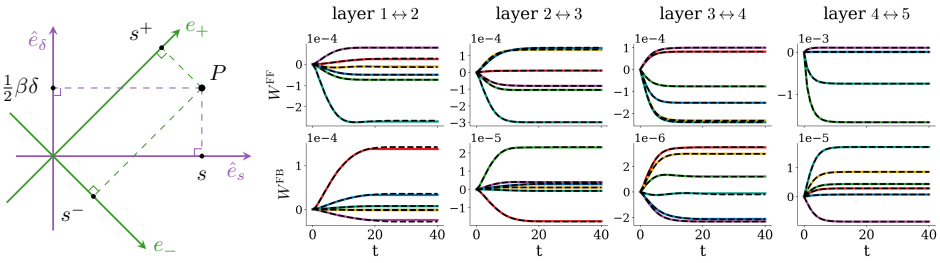
where the last two terms vanish by equation 3. Consequently the gradient of the cost C with respect to θ can be computed as the gradient of $\mathcal{L}$ when s and δ are at equilibrium.

The standard sequential forward and backward pass of BP can be viewed as a special case of RBP in which the connectivity is restricted to be feedforward and the traversal order is chosen such that states and errors converge as quickly as possible. We walk through this in Appendix A

3 Recurrent BP in Dyadic Coordinates

We introduce a change of variable, which we apply to the Lagrangian in equation 2. As illustrated in Figure 1a this corresponds to represent the same point P in a different coordinate system. When $\alpha = \frac{1}{2}$ the new system is rotated $45°$ relative to the original coordinate system. We require $\beta > 0$ and $\alpha \in [0, 1]$ and define $\bar{\alpha} = 1 - \alpha$.

$$s^+ = s + \bar{\alpha}\beta\delta, \quad s^- = s - \alpha\beta\delta \qquad \Longleftrightarrow \qquad \delta = \tfrac{1}{\beta}(s^+ - s^-), \quad s = \alpha s^+ + \bar{\alpha} s^-. \tag{7}$$



(a) (b)

Figure 1: Dyadic coordinates and gradient estimates. **(a)** One state point P in the state/costate plane, read in the orthonormal primal/adjoint frame $(s, \frac{1}{2}\beta\delta)$ with unit vectors $\hat{e}_s, \hat{e}_\delta$ and in the orthogonal but non-normalized dyadic frame (s^+, s^-) with $e_\pm = (1, \pm 1)$. In this plot we have made the choice $\alpha = 1/2$. **(b)** Temporal evolution of gradient estimates for a randomly chosen subset of tensor elements during the supervised/nudged phase of learning for a randomly initialized 5 layer CNN. The solid lines indicate dyadic RBP and the dashed lines are the standard RBP gradient estimates. This experiment uses asymmetric connectivity, $\alpha = 1/2$, $\beta = 1$ and Float32 precision.

3.1 Equivalence Between Adjoint and Dyadic RBP Dynamics

We perform the change of variables given in Equation 7 on the adjoint Lagrangian equation 2 and denote the Lagrangian as $\mathcal{L}^\pm$ when it is parametrized in terms of s^+ and s^-.

$$\begin{aligned} \mathcal{L}^\pm(\theta, s^+, s^-; \beta) &= \mathcal{L}(\theta, \alpha s^+ + \bar{\alpha} s^-, \tfrac{1}{\beta}(s^+ - s^-)) \\ &= C(\alpha s^+ + \bar{\alpha} s^-) + \frac{1}{\beta}(s^+ - s^-)^\top F(\theta, \alpha s^+ + \bar{\alpha} s^-) \end{aligned} \tag{8}$$

Theorem 3.1 shows an optimization scheme for s^+ and s^- that exactly recovers the traditional root finding dynamics for s and δ in $\mathcal{L}$. We refer to RBP as dyadic RBP when expressed in the dyadic coordinates.

Theorem 3.1 (Exact equivalence of adjoint and dyadic dynamics)**.** *Let $\mathcal{L}^{\pm}(\theta, s^+, s^-; \beta)$ (equation 8) denote the Lagrangian equation 2 in the dyadic coordinates equation 7. Furthermore let s^+, s^- evolve according to the coupled dynamics*

$$\tau \begin{pmatrix} \dot{s}^+ \\ \dot{s}^- \end{pmatrix} = -\beta \begin{pmatrix} 2\bar{\alpha}I & (\bar{\alpha}-\alpha)I \\ (\bar{\alpha}-\alpha)I & -2\alpha I \end{pmatrix} \begin{pmatrix} \partial \mathcal{L}^{\pm}/\partial s^+ \\ \partial \mathcal{L}^{\pm}/\partial s^- \end{pmatrix}, \tag{9}$$

Then the dyadic system exactly recovers the adjoint systems dynamics (equation 4 and equation 5).

$$\tau(\alpha \dot{s}^+ + \bar{\alpha}\dot{s}^-) = \tau \dot{s}, \qquad \frac{\tau}{\beta}(\dot{s}^+ - \dot{s}^-) = \tau \dot{\delta} \tag{10}$$

This means the two systems share the same equilibria at which $dC/d\theta = \partial \mathcal{L}/\partial \theta = \partial \mathcal{L}^{\pm}/\partial \theta$.

Corollary 3.2 (Special cases)**.** ***pure descent/ascent:*** *When $\alpha = 1/2$ then $\bar{\alpha} = 1 - \alpha = 1/2$ as well and the dynamics in equation 9 simplify to descent in s^+ and ascent in s^-.* ***One unperturbed and one perturbed state*** *When $\alpha = 0$ then $s = \alpha s^+ + \bar{\alpha} s^- = s^-$ and $\tau \dot{s}^- = \tau \dot{s}$. Likewise when $\alpha = 1$ then $s = \alpha s^+ + \bar{\alpha} s^- = s^+$ and $\tau \dot{s}^+ = \tau \dot{s}$.*

The proof of Theorem 3.1 is presented in Appendix B. Although we arrive at our dyadic RBP Lagrangian in a different way (by way of RBP) the objective is closely related to the dyadic objective employed in (Scurria et al., 2026). The key difference is that we consider any convex combination of states s^+ and s^-. The case $\alpha \in \{0, 1\}$ is especially interesting as it avoids the need to compute the weighted average, which may be advantageous on certain hardware. In the following section we consider an alternative way to avoid evaluating the constraint F at the mid-point.

Figure 1b demonstrates that the RBP gradient estimates computed in dyadic coordinates matches the gradient estimate computed by RBP in the state/error coordinates at any step of the relaxation process.

3.2 Split Dyadic RBP

In the previous section we established that dyadic RBP is equivalent to standard RBP, as it is the same algorithm computed in a different coordinate system. In this section we will show a surrogate of the dyadic Lagrangian, which in the limit of small β approximates the adjoint dynamics, and in the special case of $\alpha \in \{0, 1\}$ recovers them exactly. This formulation allows us to only evaluate the constraints at s^+ and s^- rather than at their weighted average. Similar splitting procedures to the one we use here is employed in appendix F of (Scurria et al., 2026) for Hopfield like constraints and in appendix G of (Scurria, 2026) for feedforward constraints, though both of these assume $\alpha = 1/2$ and evaluate the original (non-split) objective when computing the weight updates. Our contribution here is to generalize the splitting procedure to general constraints F and to any $\alpha \in [0, 1]$. We also show that the error in both state dynamics, and weight gradient estimate scales with $\bar{\alpha}\alpha\beta^2$ (and hence vanishes at $\alpha \in \{0, 1\}$).

We employ the following surrogate for equation 8

$$\begin{aligned} \hat{\mathcal{L}}^{\pm}(\theta, s^+, s^-; \beta) :=& \alpha \mathcal{L}\Big(\theta, s^+, \tfrac{1}{\beta}(s^+ - s^-)\Big) + \bar{\alpha}\, \mathcal{L}\Big(\theta, s^-, \tfrac{1}{\beta}(s^+ - s^-)\Big) \\ =& \alpha \left(C(s^+) + \frac{1}{\beta}(s^+ - s^-)^\top F(\theta, s^+) \right) + \bar{\alpha} \left(C(s^-) + \frac{1}{\beta}(s^+ - s^-)^\top F(\theta, s^-) \right). \end{aligned} \tag{11}$$

Two things are worth noticing. First of all where equation 8 evaluates constraints at the weighted average $\alpha s^+ + \bar{\alpha} s^-$, the *split* dyadic RBP surrogate Lagrangian equation 11 only evaluates constraints at s^+ and s^- individually. Secondly with $\alpha \in \{0, 1\}$ equation 11 is identical to equation 8, and hence Theorem 3.1 applies and the $\dot{s}$ and $\dot{\delta}$ dynamics are recovered exactly. Consequently the interesting question is not what happens when α is neither 0 or 1 but something in between. Theorem 3.3 states the optimization dynamics for this objective and the approximate equivalence to the RBP dynamics.

Theorem 3.3 (Approximate equivalence of adjoint and split dyadic dynamics)**.** *Assume C and F are three times continuously differentiable in s, let $\alpha \in [0, 1]$, $\beta > 0$, and let $\hat{\mathcal{L}}^{\pm}$ be the split dyadic Lagrangian of Equation 11. Let s^+, s^- evolve according to*

$$\tau \begin{pmatrix} \dot{s}^+ \\ \dot{s}^- \end{pmatrix} = -\beta \begin{pmatrix} 2\bar{\alpha}I & (\bar{\alpha}-\alpha)I \\ (\bar{\alpha}-\alpha)I & -2\alpha I \end{pmatrix} \begin{pmatrix} \partial \hat{\mathcal{L}}^{\pm}/\partial s^+ \\ \partial \hat{\mathcal{L}}^{\pm}/\partial s^- \end{pmatrix}, \tag{12}$$

and write $s = \alpha s^+ + \bar{\alpha} s^-$ *and* $\delta = \frac{1}{\beta}(s^+ - s^-)$*. Then, for fixed* (s, δ) *as* $\beta \to 0$*,*

$$\tau(\alpha \dot{s}^+ + \bar{\alpha} \dot{s}^-) = -F(\theta, s) + \mathcal{O}(\alpha\bar{\alpha}\beta^2), \tag{13}$$

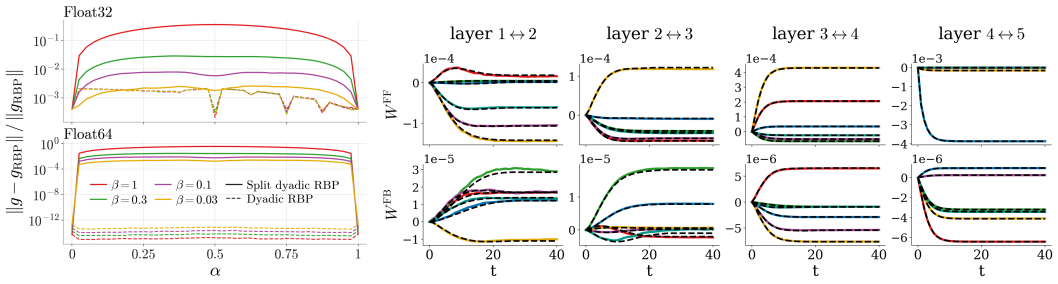
$$\tfrac{\tau}{\beta}(\dot{s}^+ - \dot{s}^-) = -G(\theta, s, \delta) + \mathcal{O}(\alpha\bar{\alpha}\beta^2), \tag{14}$$

matching the adjoint dynamics equation 4 and equation 5 up to $\mathcal{O}(\alpha\bar{\alpha}\beta^2)$*. Furthermore, the gradient of* $\hat{\mathcal{L}}^{\pm}$ *with respect to the weights matches the gradient of* $\mathcal{L}$ *at any point* (s^+, s^-) *up to* $\mathcal{O}(\alpha\bar{\alpha}\beta^2)$

$$\frac{\partial \hat{\mathcal{L}}^{\pm}(\theta, s^+, s^-; \beta)}{\partial \theta} = \frac{\partial \mathcal{L}(\theta, s, \delta)}{\partial \theta} + \mathcal{O}(\alpha\bar{\alpha}\beta^2). \tag{15}$$

This result may come as a surprise to researchers from the related equilibrium propagation literature, where symmetric nudging is used to reduce the approximation error to $\mathcal{O}(\beta^2)$ (Laborieux et al., 2021). In split dyadic RBP on the other hand the approximation error is $\mathcal{O}(\bar{\alpha}\alpha\beta^2)$, meaning it entirely vanishes in the setting $\alpha\{0, 1\}$ and is the worst in the symmetric nudging setting $\alpha = 1/2$.

Figure 2a compares the split dyadic RBP and the dyadic RBP gradient estimates to the standard RBP estimate as a function of α for different choices of β. For dyadic RBP the error is purely due to numerical precision. For split dyadic RBP the error shows a clear β dependence for $\alpha \notin \{0, 1\}$, and for $\alpha \in \{0, 1\}$ split dyadic RBP's error drops down to the same level as that of dyadic RBP. Figure 2b shows the gradient alignment between split dyadic RBP and RBP across time steps. Compared to Figure 1b the alignment is slightly worse here as split dyadic RBP only approximates RBP when $\alpha \notin \{0, 1\}$.



(a) (b)

Figure 2: **(a)** Relative error of the weight gradient for dyadic RBP and split dyadic RBP compared to standard RBP, as a function of α, for various choices of β and in both Float32 (top) and Float64 (bottom). **(b)** Temporal evolution of gradient estimates for a randomly chosen subset of tensor elements during the supervised/nudged phase of learning for a randomly initialized 5 layer CNN. The solid lines indicate split dyadic RBP and the dashed lines are the standard RBP gradient estimates. This experiment uses asymmetric connectivity, $\alpha = 1/2$, $\beta = 0.01$ and Float32 precision.

4 Empirical Validation

We validate this work along three axes: (i) the fidelity of dyadic gradient estimates against exact BPTT is explored for $\alpha\{0, 1/2, 1\}$ in Appendix E), (ii) classification accuracy compared to a feedforward baseline (traditional "backprop") , and (iii) an image-denoising task where we both employ a U-Net inspired architecture as well as a feedback driven encoder- network which leverages feedback from deeper layers to denoise at the first hidden layer. Full experimental details are given in Appendix D.

4.1 Classification Experiments

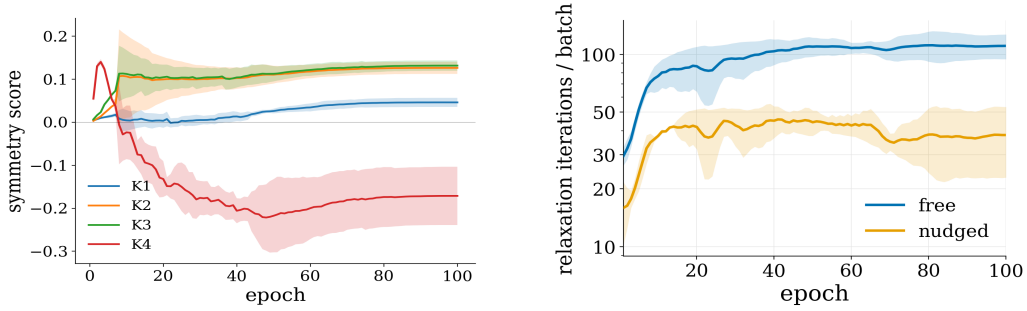
Table 1 reports CIFAR-10 test accuracy for the 5-layer CNN under feedforward and asymmetric connectivity. For each connectivity we compare RBP with both dyadic RBP and split dyadic RBP. Split Dyadic RBP is evaluated at both alpha=0.5 and alpha=1. At $\alpha = 1$ it is equivalent to RBP and consequently insensitive to nudging strength training successfully with a large nudging gain of

$\beta = 10$ (see Appendix F). In all variations dyadic RBP and split dyadic RBP perform equivalently to RBP, consistent with our theoretical results. Since the asymmetric network has feedback connections it has twice as many parameters as the feedforward network, even though they have the same number of neurons. For this reason we add a parameter-matched asymmetric control whose channel widths are reduced to match the feedforward model, disentangling parameter count and connectivity.

Table 1: CIFAR-10 test accuracy (%) for the 5-layer CNN. RBP vs. dyadic RBP vs. split dyadic RBP, under feedforward and asymmetric connectivity. Accuracies are mean ± std over 3 seeds. The final column matches the asymmetric parameter count to the feedforward model.

Algorithm	Test acc. (%)		
	Feedforward (15.43M)	Asymmetric (30.84M)	Asym. (matched) (15.45M)
RBP	91.75 ± 0.09	92.59 ± 0.15	92.37 ± 0.07
Dyadic RBP ($\alpha = 0.5, \beta = 1$)	91.68 ± 0.05	92.62 ± 0.18	92.28 ± 0.23
Split Dyadic RBP (α=0.5, β=0.01)	91.60 ± 0.14	92.52 ± 0.14	92.43 ± 0.12
Split Dyadic RBP (α=1, β=10)	91.71 ± 0.02	92.83 ± 0.10	92.48 ± 0.12

Feedback symmetry and inference cost. Although the feedback weights and feedforward weights are independent in the asymmetric networks, training induces some structure between each feedback kernel and the transpose of its forward counterpart. Figure 3a tracks this per-layer symmetry score (defined in Equation 19 in appendix D.3) over training: the score departs from zero and stabilizes, with a consistent sign pattern across layers, under both energy forms. Figure 3b reports the number of relaxation (fixed-point) iterations per batch consumed by the free and nudged phases under asymmetric connectivity.



(a) Feedback/forward symmetry

(b) Split Dyadic-RBP mean iterations

Figure 3: Asymmetric 5-layer CNN over 100 epochs of split-dyadic training (CIFAR-10;beta=0.01,alpha=0.5; mean ± std over seeds). **(a)** Per-layer symmetry score between each feedback kernel and the transpose of its forward counterpart (split dyadic RBP); zero indicates uncorrelated feedback. **(b)** Per-batch mean relaxation iterations in the free and nudged phases.

4.2 Denoising Experiments

We finally evaluate the formulation on image denoising, on both FASHION-MNIST and CIFAR-10. For full results see Appendix G. As Table 9) shows, across depth and both reconstruction sites (Table 2) the asymmetric and feedforward networks denoise reliably, whereas symmetric and skew-symmetric connectivity are prone to failure when the reconstruction is read at the output. Figure 6 shows representative CIFAR-10 test reconstructions. Full results are presented in Appendix G.

Table 2: Denoising results: The raw summed test MSE for Fashion-MNIST and Cifar-10, with best depth in parentheses. Encoder-only reads the reconstruction at the input, the U-Net (encoder–decoder) at the output. Feedforward encoder-only architecture is skipped as it can not exploit feedback to denoise the first layer. Symmetric and skew-symmetric connectivity are worst at the output, and on CIFAR-10 they denoise at depth 1 but destabilize with depth (fail by depth 3–7). Asymmetric is strong at both sites, and is the only connectivity that keeps improving with depth: the CIFAR-10 asymmetric U-Net scales $35.8\,(d1) \to 32.4\,(d3) \to 31.2\,(d7) \to 28.8\,(d9)$.

Dataset	Connectivity	Encoder-only (in)	U-Net (out)
FASHION-MNIST	Asymmetric	12.3 ± 0.2 (d3)	10.4 ± 0.2 (d3)
	Symmetric	12.5 ± 0.1 (d3)	16.4 ± 1.6 (d1)
	Skew-sym.	15.4 ± 0.1 (d3)	13.4 ± 0.1 (d3)
	Feedforward	—	10.8 ± 0.0 (d3)
CIFAR-10	Asymmetric	30.1 ± 0.3 (d8)	28.8 ± 0.2 (d9)
	Symmetric	37.3 ± 0.2 (d3)	40.9 ± 0.4 (d1)
	Skew-sym.	37.1 ± 0.0 (d3)	37.3 ± 0.4 (d1)
	Feedforward	—	29.7 ± 0.5 (d7)

5 Discussion

As we have demonstrated, many of the concerns presented in the literature on bio-plausible learning paradigms can be met by recurrent backpropagation alone, which possesses *all* the locality properties of equilibrium propagation while also being applicable to asymmetric networks. An important consideration seldom discussed in the literature on bio-plausibility (most biologically inspired learning algorithms having only been explored in the context of feedforward and symmetric networks) is that connectivity in the brain is not likely constrained in the ways that several previously proposed solutions would require.

While the computational importance of recurrence has long been emphasized in models of thalamocortical activity (Sompolinsky et al. (1988); Izhikevich & Edelman (2008); Sussillo & Abbott (2009)Douglas et al. (1995); Markov et al. (2014)), asymmetric connectivity (Song et al. (2005); Wagner et al. (2026)) and feedback in general (Hupé et al. (1998); Fişek et al. (2023); Sepulchre (2022); Tugsbayar et al. (2026)), are also known to have an important role in biology. These observations motivate learning frameworks that accommodate connectivity-agnostic recurrent computation. Our denoising task shows directly that asymmetric connectivity offers the advantage of being relatively agnostic to problem setting compared with feedforward and symmetric/skew-symmetric connectivity.

Related Works A requirement of training on asymmetric networks is actual convergence to equilibria. Due to dynamic instability, it is in practice necessary to have a tolerance-based convergence criterion D.3 as well as the step-size damping heuristic H discussed in the appendix. Though it does not specifically explore asymmetric connectivity, or biological plausibility, some of the challenges arising in training asymmetric recurrent networks are shared by the literature on Deep Equilibrium Models (Bai et al. (2019; 2021)). In particular, work on monotone operators (Winston & Kolter (2020)), which explicitly parametrize networks via skew and symmetric components, offers convergence guarantees for such settings. It is possible that approaches might be borrowed from these areas to further improve the stability of training in a framework such as ours. For our purposes the heuristic approach (dynamic damping of step-sizes) discussed in H proves sufficient.

Along similar lines, *AntisymmetricRNN* (Chang et al., 2019), which employ a purely skew-symmetric connectivity, argue for enhanced stability, and better performance in spatiotemporal tasks than traditional RNNs. The notion that symmetric and skew-symmetric connectivity may be systematically combined to achieve stability in spatiotemporal or trajectory representation, in fact has biological support (Wagner et al. (2026)).

The question of asymmetric feedback has been approached in the vector-field variant of equilibrium propagation (Scellier et al., 2018), though the resulting algorithm no longer approximates the true gradient of the cost function. Follow-up works have sought to examine conditions for stable infer-

ence and and gradient alignment in vector-field EP (Farinha et al., 2020; Costa & Santos, 2025). The dyadic equilibrium propagation algorithm, which overcomes the issues of vector-field variants of EP and computes exact gradients in asymmetric networks, is the closest relative to our dyadic RBP method. In fact dyadic RBP with $\alpha = 1/2$ coincides with dyadic equilibrium propagation (Scurria et al., 2026). As mentioned we find it more natural to view these algorithms as RBP in a different coordinate system, due to the mathematical equivalence to RBP, as well as the fact that the dyadic Lagrangian is not in general an energy function (since inference is not pure minimization) and the gradient estimate lacks equilibrium propagation's classical β-dependent approximation error.

Of course, one may ask if the split dyadic RBP algorithm then should be viewed as an instance of equilibrium propagation, since it does indeed introduce a β-dependent approximation error. We are not sure, but point out split dyadic RBP's approximation error is greatest in the centered setting ($\alpha = 0.5$) and vanishes entirely (limited only by machine precision) for $\alpha \in \{0, 1\}$, which is distinct from standard incarnations of EP (where centered nudging yields a better gradient estimate than forward and backwards nudging (Laborieux et al., 2021)). Finally there is a methodological distinction worth keeping in mind. In RBP, dyadic RBP and split-dyadic RBP one constructs a scalar potential (not necessarily an energy) from constraints, whereas in equilibrium propagation one typically starts from an energy function, which (we speculate) may make EP a natural choice when dealing with black-box physical systems.

Two recent works have revisited the Lagrangian formulation of backpropagation, albeit only in the context of feedforward networks. A variant of dual propagation was derived by reparametrizing the Lagrangian corresponding to a feedforward network into dyadic coordinates (Høier & Zach, 2024). A variant of predictive coding, called augmented Lagrangian predictive coding (PC-ALM), adds the Lagrangian corresponding to a feedforward network to the energy function of a feedforward predictive coding network (Seely & Gould, 2026), in order to make error signals propagate more efficiently in very deep networks. The authors discount optimizing LeCun's Lagrangian directly with the argument that ascent on the adjoint variable will not converge (A similar argument is found in (Scurria, 2026)). This criticism overlooks the fact that the optimization scheme proposed in (LeCun, 1988) does not involve ascent in the adjoint variable, but rather consists in a coupled root finding algorithm. Additionally, as Corollary 3.2 shows, this root-finding algorithm can be viewed as descent in s^+ and ascent in s^- with the choice $\alpha = 1/2$

Limitations and Future work While our experiments often show an advantage of asymmetric connectivity, we do not account for the additional inference time required to fully train a network in the recurrent (asymmetric) setting. From this perspective, the somewhat slight performance gains offered by asymmetric connectivity, are less of practical than theoretical interest. On the other hand, feedforward networks benefit implicitly from architectures and computational frameworks constructed over decades of research biased in their favor. It is probable that other task settings and configurations may exist where learned feedback is necessary, as it appears to be in the brain (Tugsbayar et al. (2026); Wagner et al. (2026)). One important potential direction of this kind of research would be the design and deliberate exploration of such settings.

A very important detail of (split) dyadic RBP is that it it does not avoid one of the most important bio-plausibility concerns discussed in the literature; namely weight transport. Indeed during inference our asymmetric networks do not require weight transport, but during training, error signals are transported via the transpose of the Jacobian of F. One could of course combine our approach with methods such feedback alignment, Kolen-Pollack learning, weight mirrors or some similar approaches. Nevertheless, the focus of this work is not to make the case that backprop is unconditionally biologically plausible, but to offer a challenge to existing literature on biological plausibility, by showing that one can make a case for RBP being *as* biologically plausible as many of the previously proposed algorithms from the literature.

Finally, an obvious extension of our work is to reparametrize into polar coordinates rather than dyadic coordinates. This means rather than (s, δ) or (s^+, s^-) one would use a magnitude r and an angle ϕ. It is possible that there are constraint functions F or specific hardware for which polar RBP would be advantageous.

6 AI USE STATEMENT

Our codebase was largely written by hand, though some features (logging utilities, plotting, code refactoring, latex formatting, bash scripts for launching experiments) were facilitated by LLMs. LLMs were used in an iterative manner for drafting and revising proofs. Additionally we asked LLMs for feedback on our writing of some sections of the manuscript. All hypotheses and experiments were designed by us. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Mohamed Akrout, Collin Wilson, Peter Humphreys, Timothy Lillicrap, and Douglas B Tweed. Deep learning without weight transport. *Advances in neural information processing systems*, 32, 2019.

Larry Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *Advances in neural information processing systems*, 32, 2019.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Stabilizing equilibrium models by jacobian regularization. *arXiv preprint arXiv:2106.14342*, 2021.

Sergey Bartunov, Adam Santoro, Blake A. Richards, Luke Marris, Geoffrey E. Hinton, and Timothy P. Lillicrap. Assessing the scalability of biologically-motivated deep learning algorithms and architectures. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

Yoshua Bengio. How auto-encoders could provide credit assignment in deep networks via target propagation. *arXiv preprint arXiv:1407.7906*, 2014.

James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

Bo Chang, Minmin Chen, Eldad Haber, and Ed H Chi. Antisymmetricrnn: A dynamical system view on recurrent neural networks. *arXiv preprint arXiv:1902.09689*, 2019.

Pedro Costa and Pedro A Santos. Directed equilibrium propagation revisited. *Mathematics*, 13(11): 1866, 2025.

Francis Crick. The recent excitement about neural networks. *Nature*, 337(6203):129–132, 1989. doi: 10.1038/337129a0.

Stephan Dempe and Alain B Zemkoho. The bilevel programming problem: reformulations, constraint qualifications and optimality conditions. *Mathematical Programming*, 138(1):447–473, 2013.

Rodney J. Douglas, Christof Koch, Misha Mahowald, Kevan A. C. Martin, and Humbert H. Suarez. Recurrent Excitation in Neocortical Circuits. *Science*, 269(5226):981–985, 1995. doi: 10.1126/science.7638624. URL `https://doi.org/10.1126/science.7638624`.

Maxence Ernoult, Julie Grollier, Damien Querlioz, Yoshua Bengio, and Benjamin Scellier. Equilibrium propagation with continual weight updates. *arXiv preprint arXiv:2005.04168*, 2020.

Matilde Tristany Farinha, Sérgio Pequito, Pedro A Santos, and Mário AT Figueiredo. Equilibrium propagation for complete directed neural networks. *arXiv preprint arXiv:2006.08798*, 2020.

Mehmet Fişek, Dustin Herrmann, Alexander Egea-Weiss, Matilda Cloves, Lisa Bauer, Tai-Ying Lee, Lloyd E. Russell, and Michael Häusser. Cortico-cortical feedback engages active dendrites in visual cortex. *Nature*, 617(7962):769–776, 2023. doi: 10.1038/s41586-023-06007-6. URL `https://doi.org/10.1038/s41586-023-06007-6`.

Charlotte Frenkel, Martin Lefebvre, and David Bol. Learning without feedback: Fixed random learning signals allow for feedforward training of deep neural networks. *Frontiers in Neuroscience*, 15:629892, 2021. doi: 10.3389/fnins.2021.629892.

Chad R Galley. Classical mechanics of nonconservative systems. *Physical review letters*, 110(17): 174301, 2013.

Jordan Guerguiev, Timothy P Lillicrap, and Blake A Richards. Towards deep learning with segregated dendrites. *Elife*, 6:e22901, 2017.

Rasmus Høier and Christopher Zach. Two tales of single-phase contrastive hebbian learning. In *Proceedings of the 41st International Conference on Machine Learning*, pp. 18470–18488. PMLR, 2024.

Rasmus Høier, D Staudt, and Christopher Zach. Dual propagation: Accelerating contrastive hebbian learning with dyadic neurons. In *International Conference on Machine Learning*, pp. 13141–13156. PMLR, 2023.

Rasmus Høier, Kirill Kalinin, Maxence Ernoult, and Christopher Zach. Dyadic learning in recurrent and feedforward models. In *NeurIPS 2024 Workshop Machine Learning with new Compute Paradigms*, 2024. URL https://openreview.net/pdf?id=LNfWowAErI.

John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, 81(10):3088–3092, 1984.

J. M. Hupé, A. C. James, B. R. Payne, S. G. Lomber, P. Girard, and J. Bullier. Cortical feedback improves discrimination between figure and background by V1, V2 and V3 neurons. *Nature*, 394 (6695):784–787, 1998. doi: 10.1038/29537. URL https://doi.org/10.1038/29537.

Eugene M. Izhikevich and Gerald M. Edelman. Large-scale model of mammalian thalamocortical systems. *Proceedings of the National Academy of Sciences*, 105(9):3593–3598, 2008. doi: 10.1073/pnas.0712231105. URL https://doi.org/10.1073/pnas.0712231105.

Max Jaderberg, Wojciech Marian Czarnecki, Simon Osindero, Oriol Vinyals, Alex Graves, David Silver, and Koray Kavukcuoglu. Decoupled neural interfaces using synthetic gradients. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1627–1635. PMLR, 2017.

John F. Kolen and Jordan B. Pollack. Backpropagation without weight transport. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, volume 3, pp. 1375–1380. IEEE, 1994. doi: 10.1109/ICNN.1994.374486.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Axel Laborieux, Maxence Ernoult, Benjamin Scellier, Yoshua Bengio, Julie Grollier, and Damien Querlioz. Scaling equilibrium propagation to deep convnets by drastically reducing its gradient estimator bias. *Frontiers in neuroscience*, 15:129, 2021.

Yann LeCun. A theoretical framework for back-propagation. In *Proceedings of the 1988 connectionist models summer school*, volume 1, pp. 21–28, 1988.

Dong-Hyun Lee, Saizheng Zhang, Asja Fischer, and Yoshua Bengio. Difference target propagation. In *Machine Learning and Knowledge Discovery in Databases (ECML PKDD 2015)*, volume 9284 of *Lecture Notes in Computer Science*, pp. 498–515. Springer, 2015. doi: 10.1007/978-3-319-23528-8_31.

Qianli Liao, Joel Z. Leibo, and Tomaso A. Poggio. How important is weight symmetry in backpropagation? In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pp. 1837–1844, 2016. doi: 10.1609/aaai.v30i1.10279.

Timothy P. Lillicrap, Daniel Cownden, Douglas B. Tweed, and Colin J. Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7: 13276, 2016. doi: 10.1038/ncomms13276.

Timothy P Lillicrap, Adam Santoro, Luke Marris, Colin J Akerman, and Geoffrey Hinton. Backpropagation and the brain. *Nature Reviews Neuroscience*, 21(6):335–346, 2020.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

N. T. Markov, M. M. Ercsey-Ravasz, A. R. Ribeiro Gomes, C. Lamy, L. Magrou, J. Vezoli, P. Misery, A. Falchier, R. Quilodran, M. A. Gariel, J. Sallet, R. Gamanut, C. Huissoud, S. Clavagnier, P. Giroud, D. Sappey-Marinier, P. Barone, C. Dehay, Z. Toroczkai, K. Knoblauch, D. C. Van Essen, and H. Kennedy. A Weighted and Directed Interareal Connectivity Matrix for Macaque Cerebral Cortex. *Cerebral Cortex*, 24(1):17–36, 2014. doi: 10.1093/cercor/bhs270. URL https://doi.org/10.1093/cercor/bhs270.

Alexander Meulemans, Francesco Carzaniga, Johan Suykens, João Sacramento, and Benjamin F. Grewe. A Theoretical Framework for Target Propagation. In *Advances in Neural Information Processing Systems*, volume 33, pp. 20024–20036. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/e7a425c6ece20cbc9056f98699b53c6f-Paper.pdf.

Alexander Meulemans, Matilde Tristany Farinha, Javier Garcia Ordonez, Pau Vilimelis Aceituno, João Sacramento, and Benjamin F. Grewe. Credit Assignment in Neural Networks through Deep Feedback Control. In *Advances in Neural Information Processing Systems*, volume 34, pp. 4674–4687. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/25048eb6a33209cb5a815bff0cf6887c-Paper.pdf.

Beren Millidge, Anil Seth, and Christopher L Buckley. Predictive coding: a theoretical and experimental review. *arXiv preprint arXiv:2107.12979*, 2021.

Javier R Movellan. Contrastive hebbian learning in the continuous hopfield model. In *Connectionist models*, pp. 10–17. Elsevier, 1991.

Arild Nøkland. Direct feedback alignment provides learning in deep neural networks. *Advances in neural information processing systems*, 29, 2016.

Randall C O'Reilly. Biologically plausible error-driven learning using local activation differences: The generalized recirculation algorithm. *Neural computation*, 8(5):895–938, 1996.

Jiří V Outrata. A note on the usage of nondifferentiable exact penalties in some special optimization problems. *Kybernetika*, 24(4):251–258, 1988.

Alexandre Payeur, Jordan Guerguiev, Friedemann Zenke, Blake A. Richards, and Richard Naud. Burst-dependent synaptic plasticity can coordinate learning in hierarchical circuits. *Nature Neuroscience*, 24:1010–1019, 2021. doi: 10.1038/s41593-021-00857-x.

Rajesh P. N. Rao and Dana H. Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999. doi: 10.1038/4580.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

João Sacramento, Rui Ponte Costa, Yoshua Bengio, and Walter Senn. Dendritic cortical microcircuits approximate the backpropagation algorithm. *Advances in neural information processing systems*, 31, 2018.

Tommaso Salvatori, Yuhang Song, Yordan Yordanov, Beren Millidge, Lei Sha, Cornelius Emde, Zhenghua Xu, Rafal Bogacz, and Thomas Lukasiewicz. A stable, fast, and fully automatic learning algorithm for predictive coding networks. In *International Conference on Learning Representations*, volume 2024, 2024.

Benjamin Scellier and Yoshua Bengio. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation. *Frontiers in computational neuroscience*, 11:24, 2017.

Benjamin Scellier, Anirudh Goyal, Jonathan Binas, Thomas Mesnard, and Yoshua Bengio. Generalization of equilibrium propagation to vector field dynamics. *arXiv preprint arXiv:1808.04873*, 2018.

Antonino Emanuele Scurria. A physical theory of backpropagation: Exact gradients from the least-action principle. *arXiv preprint arXiv:2602.02281*, 2026.

Antonino Emanuele Scurria, Dimitri Vanden Abeele, Bortolo Matteo Mognetti, and Serge Massar. Equilibrium propagation for non-conservative systems. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=5785emCgS1.

Jeffrey Seely and Julian Gould. Augmented lagrangian predictive coding. *arXiv preprint arXiv:2605.31022*, 2026.

Rodolphe Sepulchre. Spiking Control Systems. *Proceedings of the IEEE*, 110(5):577–589, 2022. doi: 10.1109/jproc.2022.3163926. URL https://arxiv.org/abs/2112.03565.

H. Sompolinsky, A. Crisanti, and H. J. Sommers. Chaos in Random Neural Networks. *Physical Review Letters*, 61(3):259–262, 1988. doi: 10.1103/physrevlett.61.259. URL https://doi.org/10.1103/physrevlett.61.259.

Sen Song, Per Jesper Sjöström, Markus Reigl, Sacha Nelson, and Dmitri B Chklovskii. Highly Nonrandom Features of Synaptic Connectivity in Local Cortical Circuits. *PLoS Biology*, 3(3):e68, 2005. doi: 10.1371/journal.pbio.0030068. URL https://doi.org/10.1371/journal.pbio.0030068.

David Sussillo and L.F. Abbott. Generating Coherent Patterns of Activity from Chaotic Neural Networks. *Neuron*, 63(4):544–557, 2009. doi: 10.1016/j.neuron.2009.07.018. URL https://doi.org/10.1016/j.neuron.2009.07.018.

Mashbayar Tugsbayar, Mingze Li, Eilif B Muller, and Blake Richards. Top-down feedback in deep neural networks leads to functional differences during audiovisual integration. *eLife*, 14: RP105953, 2026. doi: 10.7554/eLife.105953.3. URL https://elifesciences.org/articles/105953. Reviewed preprint, version 3.

Robert Urbanczik and Walter Senn. Learning by the dendritic prediction of somatic spiking. *Neuron*, 81(3):521–528, 2014. doi: 10.1016/j.neuron.2013.11.030.

Margot Wagner, Yusi Chen, Arjun Karuvally, Mia Cameron, and Terry Sejnowski. Balancing stability and flow in hippocampal networks via inductive bias and learned symmetry breaking. *Philosophical Transactions B*, 381(1954):20250260, 2026. doi: 10.1098/rstb.2025.0260. URL https://doi.org/10.1098/rstb.2025.0260.

James CR Whittington and Rafal Bogacz. An approximation of the error backpropagation algorithm in a predictive coding network with local hebbian synaptic plasticity. *Neural computation*, 29(5): 1229–1262, 2017.

James CR Whittington and Rafal Bogacz. Theories of error back-propagation in the brain. *Trends in cognitive sciences*, 23(3):235–250, 2019.

Ezra Winston and J Zico Kolter. Monotone operator equilibrium networks. *Advances in neural information processing systems*, 33:10718–10728, 2020.

Will Xiao, Honglin Chen, Qianli Liao, and Tomaso Poggio. Biologically-plausible learning algorithms can scale to large datasets. In *International Conference on Learning Representations*, 2019.

Xiaohui Xie and H. Sebastian Seung. Equivalence of backpropagation and contrastive Hebbian learning in a layered network. *Neural Computation*, 15(2):441–454, 2003. doi: 10.1162/089976603762552988.

Su-in Yi, Jack D Kendall, R Stanley Williams, and Suhas Kumar. Activity-difference training of deep neural networks using memristor crossbars. *Nature Electronics*, 6(1):45–51, 2023.

A Backpropagation in Feedforward Models as a Special Case of Recurrent Backpropagation

Assume we are dealing with a feedforward network and that the state vector s is ordered with the topology in mind ($s^\top = [s_1^\top, s_2^\top, ...]$). Assume that $F(\theta, s) = s - f(\theta_{\text{hidden}} s + \theta_{\text{input}} x)$, where f is a block structured activation function such that f_i defines the activation function applied to layer i. The trainable parameters are $\theta = \{\theta_{\text{input}}, \theta_{\text{hidden}}\}$, where θ_{hidden} is a block structured matrix defining the interactions between layers and θ_{input} is a matrix that projects a fixed input $s_0 := x$ to the first hidden layer. For simplicity we consider the special case of a feedforward network without skip-connections, which means θ_{hidden} only has non-zero blocks on the first block subdiagonal. Then the Jacobian $\partial F/\partial s = I - \operatorname{diag}(f'(\cdot))\,\theta_{\text{hidden}}$ is lower uni-triangular (zeros above main diagonal and ones on main diagonal), ensuring the invertibility we required in section 2.

$$F(\theta, s) = s - f(\theta_{\text{hidden}} s + \theta_{\text{input}}\, x), \quad \theta_{\text{hidden}} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ W_2 & 0 & \cdots & 0 & 0 \\ 0 & W_3 & \ddots & & 0 \\ \vdots & & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & W_L & 0 \end{bmatrix}, \quad \theta_{\text{input}} = \begin{bmatrix} W_1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

We choose $\tau = 1$, initialize all $s_i = 0$ (a vector of zeroes), and solve the s dynamics of equation 4 with Euler steps with step size 1. We likewise solve the adjoint dynamics of equation 5 with Euler steps of step size 1.

$$s_i \leftarrow s_i - [F(\theta, s)]_i = f_i\left(W_i s_{i-1}\right)$$

$$\delta_i \leftarrow \delta_i - \left(\delta_i - W_{i+1}^\top \left(f'_{i+1}\left(W_{i+1} s_i\right) \odot \delta_{i+1}\right) + \frac{\partial C(s)}{\partial s_i}\right) \quad (i < L), \quad \delta_L \leftarrow \delta_L - \left(\delta_L + \frac{\partial C(s)}{\partial s_L}\right)$$

It is worth pointing out that at the fixed point $\delta_L = -\partial C/\partial s_L$ and, more generally, δ equals the negative of the error signal in the conventional presentation of BP. This is due to the sign convention in equation 2 and does not affect the validity of the gradient estimate $\partial \mathcal{L}/\partial \theta = \delta^\top \partial F/\partial \theta$.

Notice that states s_i only depends on states s_{i-1} and do not depend on the adjoint variables. Likewise δ_i only depends on the states s and on adjoint variables δ_{i+1}. Consequently the most efficient solution is to first solve for s by traversing the layers from input to output (the forward pass) and subsequently solve for δ by traversing layers from output to input (the backward pass), which is the standard backpropagation traversal in a feedforward network. Once converged the gradient of the cost with respect to the weights is just the gradient of the Lagrangian $\partial \mathcal{L}/\partial W_i = -\left(f_i'(W_i s_{i-1}) \odot \delta_i\right) s_{i-1}^\top$.

B Proof of Theorem 3.1

equation 9 gives us the following dynamics for the variables s^+ and s^-.

$$\tau \dot{s}^+ = -F(\theta, \alpha s^+ + \bar{\alpha} s^-) - \bar{\alpha}\left(\beta \frac{\partial C(\alpha s^+ + \bar{\alpha} s^-)}{\partial s} + \left(\frac{\partial F(\theta, \alpha s^+ + \bar{\alpha} s^-)}{\partial s}\right)^\top (s^+ - s^-)\right),$$

$$\tau \dot{s}^- = -F(\theta, \alpha s^+ + \bar{\alpha} s^-) + \alpha\left(\beta \frac{\partial C(\alpha s^+ + \bar{\alpha} s^-)}{\partial s} + \left(\frac{\partial F(\theta, \alpha s^+ + \bar{\alpha} s^-)}{\partial s}\right)^\top (s^+ - s^-)\right),$$

Using the variable substitution $s = \alpha s^+ + \bar{\alpha} s^-$ and $\delta = \frac{1}{\beta}(s^+ - s^-)$ (equation 7) we then get

$$\tau(\alpha \dot{s}^+ + \bar{\alpha} \dot{s}^-) = -F(\theta, \alpha s^+ + \bar{\alpha} s^-) = \tau \dot{s}$$

and

$$\begin{aligned} \frac{\tau}{\beta}(\dot{s}^+ - \dot{s}^-) &= -\frac{1}{\beta}\left(\beta \frac{\partial C(\alpha s^+ + \bar{\alpha} s^-)}{\partial s} + \left(\frac{\partial F(\theta, \alpha s^+ + \bar{\alpha} s^-)}{\partial s}\right)^\top (s^+ - s^-)\right) \\ &= -\frac{\partial C(s)}{\partial s} - \left(\frac{\partial F(\theta, s)}{\partial s}\right)^\top \delta = \tau \dot{\delta}, \end{aligned}$$

Perfectly matching Equations 4 and 5. As the temporal dynamics are equivalent relaxation in the dyadic and adjoit coordinates will converge to the same equilibria at which $dC/d\theta = \partial \mathcal{L}/\partial \theta = \partial \mathcal{L}^\pm/\partial \theta$.

C Proof of Theorem 3.3

We start out by defining a Lemma which we will use in the proof of Theorem 3.3.

Lemma C.1 (Differences and weighted sums)**.** *Let $U : \mathbb{R}^n \to \mathbb{R}$ be twice continuously differentiable, let $s, \delta \in \mathbb{R}^n$, $\alpha \in [0,1]$, $\bar{\alpha} = 1 - \alpha$, $\beta > 0$, and let $s^+ = s + \bar{\alpha}\beta\delta$ and $s^- = s - \alpha\beta\delta$. Then the weighted sum and the difference of U evaluated at s^+ and s^- satisfy*

$$\alpha U(s^+) + \bar{\alpha} U(s^-) = U(s) + \alpha\bar{\alpha}\beta^2 R_\Sigma \tag{16}$$

$$U(s^+) - U(s^-) = \beta \nabla U(s)^\top \delta + \beta^2 R_\Delta, \tag{17}$$

where $|R_\Sigma| \leq \frac{1}{2} M \|\delta\|^2$ and $|R_\Delta| \leq \frac{1}{2} M \|\delta\|^2$ with $M := \max_{|t|\leq 1} \left\| \nabla^2 U(s + t\beta\delta) \right\|_2$. Note, when U is vector or matrix-valued, we apply the lemma to each entry of U separately.

Proof of Lemma C.1 The lemma is proven using the first order Taylor expansion with integral remainder.

$$U(s+h) = U(s) + (\nabla U(s))^\top h + \int_0^1 (1-t) h^\top \nabla^2 U(s + th) h dt$$

For s^+ we plug in $\bar{\alpha}\beta\delta$ in place of h, since $s^+ = s + \bar{\alpha}\beta\delta$ and for s^- we plug in $-\alpha\beta\delta$ since $s^- = s - \alpha\beta\delta$.

$$U(s^+) = U(s) + \bar{\alpha}\beta \left(\nabla U(s)\right)^\top \delta + \bar{\alpha}^2\beta^2 \underbrace{\int_0^1 (1-t)\delta^\top \nabla^2 U(s + t\bar{\alpha}\beta\delta)\delta dt}_{=: V(s, \bar{\alpha}\beta\delta)}$$

$$U(s^-) = U(s) - \alpha\beta \nabla U(s)^\top \delta + \alpha^2\beta^2 \underbrace{\int_0^1 (1-t)\delta^\top \nabla^2 U(s - t\alpha\beta\delta)\delta dt}_{=: V(s, -\alpha\beta\delta)}$$

The weighted mean then is

$$\begin{aligned} \alpha U(s^+) + \bar{\alpha} U(s^-) &= U(s) + \alpha\bar{\alpha}\beta^2 (\bar{\alpha} V(s, \bar{\alpha}\beta\delta) + \alpha V(s, -\alpha\beta\delta)) \\ &=: U(s) + \alpha\bar{\alpha}\beta^2 R_\Sigma \end{aligned}$$

and the difference is

$$\begin{aligned} U(s^+) - U(s^-) &= \beta \nabla U(s)^\top \delta + \beta^2 (\bar{\alpha}^2 V(s, \bar{\alpha}\beta\delta) - \alpha^2 V(s, -\alpha\beta\delta)) \\ &=: \beta \nabla U(s)^\top \delta + \beta^2 R_\Delta. \end{aligned}$$

Next we bound the integral in equation 18. For $t \in [0,1]$ the point $s + t\bar{\alpha}\beta\delta$ lies on the segment $\{s + u\beta\delta : |u| \leq 1\}$ since $|t\bar{\alpha}| \leq 1$. Consequently $\|\nabla^2 U(s + t\bar{\alpha}\beta\delta)\|_2 \leq M$ by definition of M. Furthermore, by the Cauchy–Schwarz inequality and the definition of the spectral norm, $|\delta^\top A \delta| \leq \|A\|_2 \|\delta\|^2$ for any $A \in \mathbb{R}^{n\times n}$, so the integrand satisfies $(1-t)\,|\delta^\top \nabla^2 U(s + t\bar{\alpha}\beta\delta)\,\delta| \leq (1-t)\, M \|\delta\|^2$. Integrating gives

$$|V(s, \bar{\alpha}\beta\delta)| \leq M\|\delta\|^2 \int_0^1 (1-t)\, dt = \frac{1}{2} M \|\delta\|^2.$$

The same argument applied to equation 18, where the integration path $s - t\alpha\beta\delta$ also lies on the segment, gives $|V(s, -\alpha\beta\delta)| \leq \frac{1}{2} M \|\delta\|^2$. We then apply the triangle inequality to R_Σ giving

$$|R_\Sigma| \leq \bar{\alpha} |V(s, \bar{\alpha}\beta\delta)| + \alpha |V(s, -\alpha\beta\delta)| \leq \frac{1}{2} M \|\delta\|^2,$$

and apply the triangle inequality to R_Δ as well

$$|R_\Delta| \leq \bar{\alpha}^2 |V(s, \bar{\alpha}\beta\delta)| + \alpha^2 |V(s, -\alpha\beta\delta)| \leq \frac{1}{2} M \|\delta\|^2,$$

where in the last step we used that $\bar{\alpha}^2 + \alpha^2 \leq 1$. This concludes the proof of Lemma C.1.

We now proceed with the proof of Theorem 3.3 by first writing out the continuous time dynamics of s^+ and s^-.

$$\tau \dot{s}^+ = -2\bar{\alpha}\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} - (\bar{\alpha} - \alpha)\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-}$$

$$\tau \dot{s}^- = -(\bar{\alpha} - \alpha)\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} + 2\alpha\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-}$$

Next we compute the continuous time dynamics for the weighted mean of s^+ and s^-.

$$\begin{aligned}
\tau(\alpha \dot{s}^+ + \bar{\alpha}\dot{s}^-) &= -\bar{\alpha}\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} + \alpha\beta \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-} \\
&= -\bar{\alpha}\left(\alpha\beta \frac{\partial C(s^+)}{\partial s^+} + \alpha F(\theta, s^+) + \bar{\alpha} F(\theta, s^-) + \alpha \left(\frac{\partial F(\theta, s^+)}{\partial s^+}\right)^\top (s^+ - s^-)\right) \\
&\quad + \alpha\left(\bar{\alpha}\beta \frac{\partial C(s^-)}{\partial s^-} - \alpha F(\theta, s^+) - \bar{\alpha} F(\theta, s^-) + \bar{\alpha}\left(\frac{\partial F(\theta, s^-)}{\partial s^-}\right)^\top (s^+ - s^-)\right)
\end{aligned}$$

Regrouping terms we get

$$\begin{aligned}
\tau(\alpha \dot{s}^+ + \bar{\alpha}\dot{s}^-) &= -\alpha F(\theta, s^+) - \bar{\alpha} F(\theta, s^-) \\
&\quad - \alpha\bar{\alpha}\beta\left(\frac{\partial C(s^+)}{\partial s^+} - \frac{\partial C(s^-)}{\partial s^-}\right) \\
&\quad - \alpha\bar{\alpha}\left(\left(\frac{\partial F(\theta, s^+)}{\partial s^+}\right)^\top - \left(\frac{\partial F(\theta, s^-)}{\partial s^-}\right)^\top\right)(s^+ - s^-).
\end{aligned} \tag{18}$$

Applying Lemma C.1 equation 16 component-wise to the first term of equation 18 yields

$$-\alpha F(\theta, s^+) - \bar{\alpha} F(\theta, s^-) = -F(\theta, s) + \mathcal{O}(\alpha\bar{\alpha}\beta^2)$$

Applying Lemma C.1 equation 17 component-wise to the second term of equation 18 gives

$$-\alpha\bar{\alpha}\beta\left(\frac{\partial C(s^+)}{\partial s^+} - \frac{\partial C(s^-)}{\partial s^-}\right) = -\alpha\bar{\alpha}\beta\left(\beta\nabla^2 C(s)\delta + \mathcal{O}(\beta^2)\right) = \mathcal{O}(\alpha\bar{\alpha}\beta^2)$$

Let $J(s) := (\partial F(\theta, s)/\partial s)^\top$ denote the transposed Jacobian in the third term of equation 18. Applying Lemma C.1 equation 17 to each entry J_{ij} gives $J_{ij}(s^+) - J_{ij}(s^-) = \beta \nabla J_{ij}(s)^\top \delta + \mathcal{O}(\beta^2) = \mathcal{O}(\beta)$, so that $J(s^+) - J(s^-) = \mathcal{O}(\beta)$ entry-wise. Using $s^+ - s^- = \beta\delta$, the third term is therefore

$$\begin{aligned}
-\alpha\bar{\alpha}\left(J(s^+) - J(s^-)\right)(s^+ - s^-) &= -\alpha\bar{\alpha}\beta\left(J(s^+) - J(s^-)\right)\delta \\
&= -\alpha\bar{\alpha}\beta \cdot \mathcal{O}(\beta) \\
&= \mathcal{O}(\alpha\bar{\alpha}\beta^2)
\end{aligned}$$

Putting it all together we have

$$\tau(\alpha \dot{s}^+ + \bar{\alpha}\dot{s}^-) = -F(\theta, s) + \mathcal{O}(\alpha\bar{\alpha}\beta^2),$$

which proves the identity from equation 13. Next we show that the continuous time dynamics corresponding to the $1/\beta$ weighted difference approximates the continuous time dynamics of δ.

$$\begin{aligned}
\frac{\tau}{\beta}\left(\dot{s}^+ - \dot{s}^-\right) &= -2\bar{\alpha}\frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} - (\bar{\alpha} - \alpha)\frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-} + (\bar{\alpha} - \alpha)\frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} - 2\alpha\frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-} \\
&= -\frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^+} - \frac{\partial \hat{\mathcal{L}}^{\pm}}{\partial s^-} \\
&= -\left(\alpha\frac{\partial C(s^+)}{\partial s^+} + \frac{\alpha}{\beta}F(\theta, s^+) + \frac{\bar{\alpha}}{\beta}F(\theta, s^-) + \frac{\alpha}{\beta}\left(\frac{\partial F(\theta, s^+)}{\partial s^+}\right)^\top (s^+ - s^-)\right) \\
&\quad - \left(\bar{\alpha}\frac{\partial C(s^-)}{\partial s^-} - \frac{\alpha}{\beta}F(\theta, s^+) - \frac{\bar{\alpha}}{\beta}F(\theta, s^-) + \frac{\bar{\alpha}}{\beta}\left(\frac{\partial F(\theta, s^-)}{\partial s^-}\right)^\top (s^+ - s^-)\right) \\
&= -\left(\alpha\frac{\partial C(s^+)}{\partial s^+} + \bar{\alpha}\frac{\partial C(s^-)}{\partial s^-} + \frac{1}{\beta}\left(\alpha\frac{\partial F(\theta, s^+)}{\partial s^+} + \bar{\alpha}\frac{\partial F(\theta, s^-)}{\partial s^-}\right)^\top (s^+ - s^-)\right)
\end{aligned}$$

Applying Lemma C.1 equation 16 to this we get

$$\begin{aligned}\frac{\tau}{\beta}\left(\dot{s}^{+}-\dot{s}^{-}\right) &= -\frac{\partial C(s)}{\partial s}-\left(\frac{\partial F(\theta,s)}{\partial s}\right)^{\top}\delta+\mathcal{O}(\alpha\bar{\alpha}\beta^{2}) \\ &= -G(s,\delta)+\mathcal{O}(\alpha\bar{\alpha}\beta^{2}),\end{aligned}$$

Which proves equation 14.

Next we consider

$$\frac{\partial\hat{\mathcal{L}}^{\pm}(\theta,s^{+},s^{-};\beta)}{\partial\theta}=\frac{1}{\beta}\left(\alpha\frac{\partial F(\theta,s^{+})}{\partial\theta}+\bar{\alpha}\frac{\partial F(\theta,s^{-})}{\partial\theta}\right)^{\top}(s^{+}-s^{-})$$

Applying Lemma C.1 equation 16 elementwise this becomes

$$\begin{aligned}\frac{\partial\hat{\mathcal{L}}^{\pm}(\theta,s^{+},s^{-};\beta)}{\partial\theta} &= \left(\frac{\partial F(\theta,s)}{\partial\theta}+\mathcal{O}(\alpha\bar{\alpha}\beta^{2})\right)^{\top}\delta \\ &= \frac{\partial\mathcal{L}(\theta,s,\delta)}{\partial\theta}+\mathcal{O}(\alpha\bar{\alpha}\beta^{2}),\end{aligned}$$

concluding the proof.

D EXPERIMENTAL DETAILS

D.1 DATASETS

Classification and denoising experiments were both run on CIFAR-10 (Krizhevsky, 2009), consisting of 60,000 color images of size 32×32 pixels split into 10 classes (6,000 images per class), with 50,000 training and 10,000 test images. For classification, the full training set was used for training (no held-out validation split). For denoising, the training set was split 95 / 5 into 47,500 training and 2,500 validation images, with the standard 10,000-image test split used for final evaluation. Denoising experiments were additionally run on FASHION-MNIST, consisting of 60,000 training and 10,000 test grayscale images of size 28×28, zero-padded to 32×32; the same 95 / 5 train/validation split and test protocol were used.

D.2 DATA PREPROCESSING

Classification. Images were converted to $[0, 1]$, augmented, and normalized by the per-channel statistics of Table 3. Training-set augmentation consisted of (in fixed order): random horizontal flips with probability 0.5; random 32×32 crops after 4-pixel edge-replicate padding; and color jitter with brightness, contrast, and saturation factors drawn uniformly from $[0.6, 1.4]$ and hue shifts from $[-0.1, 0.1]$. Evaluation data received normalization only.

Denoising. For the denoising task images were normalized with $\mu = \sigma = 0.5$ per channel, mapping clean images to $[-1, 1]$ (matching the range of the `hard_tanh` reconstruction activation). The same training-set augmentation as in classification was applied *before* normalization (for FASHION-MNIST, augmentation is random cropping after 2-pixel padding only — no flips or color jitter), and the reconstruction target is the clean (augmented, normalized) image. Noisy inputs were constructed by adding Gaussian noise with a per-image amplitude that ramps linearly across each batch:

$$\tilde{x}_i = x_i + g_i \, \gamma \, \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, I), \qquad g_i = \tfrac{i}{n-1} + \tfrac{u_i}{n}, \quad u_i \sim \mathcal{U}[0, 1),$$

with noise factor $\gamma = 1$ and batch size n, so that every batch spans per-image noise standard deviations from ≈ 0 to $\approx \gamma$ in normalized units. Noise is drawn freshly on every pass through the data (including validation and test evaluations); noisy inputs are not clipped.

Table 3: Data normalization. Values are those used in code

Task	Dataset	Mean (μ)	Standard deviation (σ)
Classification	CIFAR-10	(0.4914, 0.4822, 0.4466)	(0.6069, 0.5982, 0.6030)
Denoising	CIFAR-10	(0.5, 0.5, 0.5)	(0.5, 0.5, 0.5)
Denoising	FASHION-MNIST	(0.5)	(0.5)

D.3 SIMULATION DETAILS

Weight initialization. Two initialization schemes were used. For consistency with previous literature, the classification CNN uses a fan-based uniform scheme ("EBL"): each forward kernel is drawn $W_{ij} \sim \mathcal{U}(-g_\ell, +g_\ell)$ with per-layer bounds $g_\ell = a_\ell / \sqrt{k^2 c_\ell}$, where $k = 3$ is the kernel size, c_ℓ the (pre-pooling) input channel count, and $a = (0.4, 0.7, 0.6, 0.3, 0.4)$ for the input projection, the three hidden convolutions, and the dense readout respectively (readout bound $0.4/\sqrt{4 \cdot 512}$). Biases are drawn $\mathcal{U}(-0.5/\sqrt{f_\ell}, +0.5/\sqrt{f_\ell})$ with f_ℓ the layer fan-in. Under asymmetric connectivity the untied feedback kernels are drawn independently with the same per-layer bounds as their forward counterparts.

The denoiser uses *Gaussian Orthogonal Ensemble* (GOE) initialization: for each (input-channel, output-channel) pair of a $N \times N$ convolution kernel, an independent symmetric $N \times N$ block is drawn as

$$W_{ij} \sim \begin{cases} \mathcal{N}\big(0, \frac{V}{N}\big), & i \neq j \\ \mathcal{N}\big(0, \frac{2V}{N}\big), & i = j \end{cases}$$

with $V = 10^{-5}$ and $N = 5$ (or $N = 1$ for the 1×1 bottleneck kernels). Denoiser biases are initialized to zero. Untied (asymmetric) feedback kernels are drawn from the same ensemble.

State initialization. All layer states are initialized to zero at the start of every free-phase relaxation, for training and evaluation alike. The nudged phase initializes both of its state copies (s^+, s^-) from the free-phase fixed point $s_\star$.

Activation functions. Hidden layers use the hard sigmoid $\sigma(x) = \max(\min(x, 1), 0)$ in both tasks. The classification output layer is linear (identity activation); the loss is applied to the raw output state. The denoiser's reconstruction layer uses $\mathrm{hard_tanh}(x) = \max(\min(x, 1), -1)$, matching the $[-1, 1]$ data range.

Loss functions. All experiments use the squared-error loss

$$\ell(o, y) = \tfrac{1}{2} \sum (o - y)^2,$$

summed over output units and batch (for classification, y is the one-hot label read from the linear output state; for denoising, y is the clean image and o the reconstruction-layer state). Reported losses are divided by the batch size, i.e. per-image half sums of squared errors; reported denoising losses are in $[-1, 1]$ pixel units (per-subpixel MSE.).

Convention to count layers. Unlike softmax-readout formulations in which the output is a function of the final hidden state but excluded from the dynamics, here the output layer is a state layer that participates fully in the relaxation. We nevertheless refer to the classification network by its four convolutional layers, counting the linear readout separately. Denoiser depths d count encoder stages: the *enc* variant adds one full-resolution reconstruction layer ($d{+}1$ state layers), the *U-Net* variant mirrors the encoder with d decoder stages ($2d$ state layers, the last being the reconstruction layer).

Classification architecture. The CNN consists of four convolutional layers and a dense readout. Every convolution is 3×3, stride 1, zero-padded (SAME); spatial downsampling is performed by a space-to-depth rearrangement (block size 2) applied *after* each convolution, which halves the spatial resolution and quadruples the channel count without discarding information (and whose adjoint, depth-to-space, is used exactly in the feedback path). Layer state shapes and kernels are listed in Table 4. Per-channel biases enter only the bottom-up drive terms; feedback terms carry no bias. Parameter counts: 15.43 M for the feedforward, symmetric and skew-symmetric models (identical parameter sets — tied feedback adds no parameters); 30.84 M for the asymmetric model (untied feedback kernels). A parameter-matched control scales the channel widths by $\approx 1/\sqrt{2}$ to $(90, 181, 362, 362)$ so that its *asymmetric* variant (15.45 M) matches, or nearly matched the tied-feedback full model's budget(within 0.14%).

Table 4: Classification CNN (CIFAR-10, space-to-depth pooling). All convolutions 3×3, stride 1, SAME padding, each followed by space-to-depth (block 2). The output layer is a linear readout excluded from the layer count by the convention used throughout this text.

Layer	State shape	Forward kernel	Feedback kernel (asym.)
input	$32 \times 32 \times 3$	—	—
hidden 1	$16 \times 16 \times 512$	$3{\times}3{\times}3{\times}128$	$3{\times}3{\times}256{\times}512$
hidden 2	$8 \times 8 \times 1024$	$3{\times}3{\times}512{\times}256$	$3{\times}3{\times}512{\times}1024$
hidden 3	$4 \times 4 \times 2048$	$3{\times}3{\times}1024{\times}512$	$3{\times}3{\times}512{\times}2048$
hidden 4	$2 \times 2 \times 2048$	$3{\times}3{\times}2048{\times}512$	10×8192
output	10	8192×10 (dense)	—

Denoising architecture. The feedback denoiser is a convolutional chain parameterized by a depth d and a base width of 32 channels. Encoder stage j has width $32 \cdot 2^j$; each stage is a downsampling stage (stride-1 SAME convolution followed by space-to-depth, block 2) while the spatial resolution exceeds 1, after which remaining stages are flat. Convolutions are 5×5 everywhere except

flat stages at 1×1 spatial resolution, which reduce to 1×1 (dense) kernels. Two variants were run: *enc* (encoder-only) prepends a full-resolution layer of shape $32 \times 32 \times 3$ at which the reconstruction is read out, followed by the d encoder stages; *U-Net* (reconstruction at the output) mirrors the d encoder stages with d decoder stages (depth-to-space followed by convolution), reads the reconstruction from the final full-resolution layer, and adds parameter-free identity skip connections between encoder and decoder layers of equal resolution. Concrete layouts and parameter counts are given in Table 5. The feedforward connectivity is run with the *U-Net* variant only, since under feedforward connectivity an input-side reconstruction layer receives no top-down drive.

Table 5: Denoiser layouts (CIFAR-10, base width 32). Downsampling by space-to-depth (block 2); 5×5 kernels except 1×1 at 1×1 spatial stages. Parameter counts: sym/skew/feedforward share the tied-feedback count; asymmetric adds untied feedback kernels. The deepest models — $d = 8$ (enc) and $d = 9$ (U-Net) — were run under asymmetric connectivity only.

Variant	d	Layer state shapes	Params (tied)	Params (asym.)
enc	1	$(32, 32, 3) \to (16, 16, 32)$	860	1,460
enc	3	$(32, 32, 3) \to (16, 16, 32) \to (8, 8, 64) \to (4, 4, 128)$	65,052	129,652
enc	7	$\cdots \to (2, 2, 256) \to (1, 1, 512) \to (1, 1, 1024) \to (1, 1, 2048)$	3,714,332	7,424,372
enc	8	$\cdots \to (1, 1, 1024) \to (1, 1, 2048) \to (1, 1, 4096)$	12,107,036	24,205,684
U-Net	1	$(16, 16, 32) \to (32, 32, 3)$	1,235	1,835
U-Net	3	$(16, 16, 32) \to (8, 8, 64) \to (4, 4, 128) \to$ mirror	129,523	258,123
U-Net	7	5 down $\to (1, 1, 1024) \to (1, 1, 2048) \to$ mirror	7,426,163	14,845,643
U-Net	9	5 down $\to (1, 1, 1024) \to \cdots \to (1, 1, 8192) \to$ mirror	91,330,675	182,636,235

Connectivity variants. All experiments use the pre-activity form of the constraint, $F(\theta, s) = s - Wf(s) - b$: each layer's state relaxes toward its *drive* $Wf(s) + b$, the affine function of the activities $f(s)$ collecting that layer's afferent input, with $\theta = (W, b)$ the block connectivity matrix and biases. Each hidden layer's drive receives bottom-up input through its forward kernel and, except under feedforward connectivity, top-down input from the layer above. Under *symmetric* connectivity, the top-down input is applied through the exact adjoint of the forward operation (transposed convolution with the spatially-flipped, channel-swapped forward kernel, composed with depth-to-space — the exact transpose of the forward map including pooling); under *skew-symmetric* connectivity it is the negative of this; under *asymmetric* connectivity a separate, independently initialized and independently trained feedback kernel of the same shape is used. In the denoiser's *U-Net* variant (reconstruction at the output), identity skip connections likewise feed back with $+1$ (symmetric) or -1 (skew-symmetric) gain; asymmetric and feedforward models have forward-only skips.

Lagrangians and algorithms. Dyadic models were trained with two-sided nudging under both Lagrangians of the main text — dyadic RBP $\mathcal{L}^{\pm}$ equation 8 and split dyadic RBP $\hat{\mathcal{L}}^{\pm}$ equation 11 — instantiated with the pre-activity constraint $F(\theta, s) = s - Wf(s) - b$ and the squared-error cost above. Unless stated otherwise $\alpha = \frac{1}{2}$; the last row of Table 1 uses $\alpha = 1$, for which the split form coincides with the dyadic form (Section 3.2). Following the dynamics equation 9 and equation 12 at $\alpha = \frac{1}{2}$, the nudged phase evolves the two state copies simultaneously, s^+ by descent and s^- by ascent on the respective Lagrangian. Both forms share the identical free phase: at $s^+ = s^- = s$ either Lagrangian reduces to descent on the constraint $F(\theta, s)$. Weight updates are the gradient of $\mathcal{L}^{\pm}/B$ (resp. $\hat{\mathcal{L}}^{\pm}/B$) with respect to the parameters at fixed states (B the batch size), with the cost term excluded (it carries no explicit parameter dependence). The recurrent-backpropagation baseline (RBP) relaxes the state/adjoint pair (s, δ) of equation 4–equation 5 jointly, with weight update $\delta^\top \partial F/\partial\theta/B$; its parameter tree is identical to the dyadic models'. By Theorem 3.1 its nudged-phase dynamics coincide exactly with those of dyadic RBP under the coordinate change equation 7.

Implementation note. Because the drive $Wf(s) + b$ is affine in the activities, the split form's per-branch constraint average collapses to a single drive evaluation, $\frac{1}{2}\big[F(\theta, s^+) + F(\theta, s^-)\big] = \bar{s} - W\bar{f} - b$ with $\bar{s} = \frac{1}{2}(s^+ + s^-)$ and $\bar{f} = \frac{1}{2}\big(f(s^+) + f(s^-)\big)$. Both Lagrangians are therefore computed by one routine differing only in where activities and cost are evaluated: $\bar{f} = f(\bar{s})$ with cost $C(\bar{s})$ for dyadic RBP, versus $\bar{f} = \frac{1}{2}(f(s^+) + f(s^-))$ with cost $\frac{1}{2}\big(C(s^+) + C(s^-)\big)$ for split dyadic RBP. This is purely a computational convenience and alters neither objective.

Root-finding algorithm. Both relaxation phases are solved by the same damped fixed-point routine, RELAX (Algorithm 1). On each iteration the layers are swept in a single sequential pass—one layer updated per color—using a forward sweep in the free phase (`traversal-free=`FORWARD) and a backward sweep in the nudged phase (`traversal-nudged=`BACKWARD); this applies to both the classification CNN and the denoiser. (A two-color odd/even schedule (ASYNC) and a fully parallel synchronous update (SYNC) are also implemented but were not used for the reported runs.) The free phase relaxes the constraint $F(\theta, s)$ to zero from a zero initialization; the nudged phase relaxes the phase-specific dynamics of each algorithm. The adjoint (RBP) and dyadic algorithms differ only in their coordinates and the per-color update handed to RELAX: Algorithm 2 relaxes a state–adjoint pair (s, δ), while Algorithm 3 relaxes a dyad (s^+, s^-) in either the dyadic or the split dyadic form.

Algorithm 1 RELAX: damped fixed-point relaxation with tolerance.

Require: per-color update U, initial state z, per-layer steps η, tolerance ε, min/max iterations $t_{\min}, T$, damping factor ρ, step floor $\eta_{\min}$, traversal colors $\mathcal{K}$

$c^{\text{prev}} \leftarrow \mathbf{1}$
for $t = 1, \ldots, T$ **do**
  $z^{\text{old}} \leftarrow z$
  for each color $k \in \mathcal{K}$ in traversal order **do**
    $z_\ell \leftarrow U_\ell(z, \eta_\ell)$ for every layer ℓ of color k ▷ FW/BW sweep
  end for
  $c_\ell \leftarrow \text{mean}_b \left\| z_\ell^{(b)} - z_\ell^{\text{old},(b)} \right\|_2 / \max\left(\|z_\ell^{\text{old},(b)}\|_2,\ 10^{-6} \right)$ for every ℓ ▷ mean relative residual
  if $t \geq t_{\min}$ **and** $\max_\ell c_\ell < \varepsilon$ **then**
    break
  end if
  $\eta_\ell \leftarrow \text{clip}\big(\eta_\ell \cdot (\rho \text{ if } c_\ell > c_\ell^{\text{prev}} \text{ else } 1),\ \eta_{\min},\ 1\big)$ for every ℓ ▷ adaptive cycle damping
  $c^{\text{prev}} \leftarrow c$
end for
return z, *hit_max* $\leftarrow (t = T)$

Algorithm 2 Adjoint RBP (state–adjoint coordinates (s, δ)).

Require: input x, parameters θ, batch size B; constraint $F(\theta, s) = s - Wf(s) - b$, cost C

Free: $s_\star \leftarrow \text{RELAX}(U^{\text{f}},\ z{=}0,\ \ldots,\ \mathcal{K}_{\text{fwd}}), \quad U_\ell^{\text{f}}:\ s_\ell \leftarrow s_\ell - \eta_\ell F_\ell(\theta, s)$
Init: $s \leftarrow s_\star, \quad \delta \leftarrow 0$
Nudged: $(s, \delta) \leftarrow \text{RELAX}(U^{\text{n}},\ (s, \delta),\ \ldots,\ \mathcal{K}_{\text{bwd}})$, with per-color update
  $s_\ell \leftarrow s_\ell - \eta_\ell F_\ell(\theta, s); \quad \delta_\ell \leftarrow \delta_\ell - \eta_\ell \big(\partial_{s_\ell} C(s) + [J_F^\top \delta]_\ell \big)$
Weight gradient: $\nabla_\theta \leftarrow \delta^\top \left(\partial F / \partial \theta \right) / B$

Algorithm 3 Dyadic RBP and split dyadic RBP (dyad coordinates (s^+, s^-)).

Require: input x, parameters θ, nudging β, batch size B, form $\in \{\text{dyadic}, \text{split}\}$

Free: $s_\star \leftarrow \text{RELAX}(U^{\text{f}},\ z{=}0,\ \ldots,\ \mathcal{K}_{\text{fwd}}), \quad U_\ell^{\text{f}}:\ s_\ell \leftarrow s_\ell - \eta_\ell F_\ell, \quad F = s - Wf(s) - b$
Init: $s^+ \leftarrow s_\star, \quad s^- \leftarrow s_\star$
Nudged: $(s^+, s^-) \leftarrow \text{RELAX}(U^{\text{n}},\ (s^+, s^-),\ \ldots,\ \mathcal{K}_{\text{bwd}})$, with per-color update
  $s_\ell^+ \leftarrow s_\ell^+ - \eta_\ell\, \partial_{s_\ell^+}(\beta \mathcal{L}); \quad s_\ell^- \leftarrow s_\ell^- + \eta_\ell\, \partial_{s_\ell^-}(\beta \mathcal{L})$
  $\mathcal{L} = \mathcal{L}^{\pm}$ equation 8 if form = dyadic, $\mathcal{L} = \hat{\mathcal{L}}^{\pm}$ equation 11 if form = split $\quad (\alpha = \frac{1}{2})$
Weight gradient: $\nabla_\theta \leftarrow (\partial_\theta\, \beta \mathcal{L}) \,/\, (\beta B)$ (cost term excluded)

Tolerance criterion and adaptive damping. Convergence is monitored through the per-layer relative state change c_ℓ of Algorithm 1: the per-example relative L_2 residual $\|\Delta z_\ell\|_2 / \max(\|z_\ell\|_2, 10^{-6})$, averaged over the batch and then maximized over layers, subject to a hard iteration cap. The free and nudged phases share this criterion. For the dyadic nudged phase the measure is taken over both branches (s^+, s^-); for the adjoint nudged phase the adjoint's change

is measured relative to the state norm $\|s\|$ rather than $\|\delta\|$, so the stopping test tracks the same quantity as the dyadic RBP criterion. If the nudged phase reaches its cap without converging, that batch's weight gradient is discarded . In practice this is indicative of an unstable configuration and does not happen in any reported runs. *Adaptive cycle damping* was enabled throughout: whenever a layer's step measure grows between consecutive iterations, its step size is multiplied by ρ (clipped to $[\eta_{\min}, 1]$ with $\eta_{\min} = 10^{-3}$), within a single relaxation only—step sizes reset every batch. In the denoising experiments the nudged phase inherits $0.95\times$ the free phase's final damped step sizes; in classification it restarts from the defaults.

Computation of symmetry for asymmetric nets. For each reciprocal pair of weight blocks — W the block carrying the forward drive from one layer to the next and U the block carrying the feedback in the reverse direction — we measure how close the pair is to the tied (symmetric) configuration $U = W^\top$ by

$$S(W,U) \;=\; \frac{\|W + U^\top\|_F^2 \;-\; \|W - U^\top\|_F^2}{\|W + U^\top\|_F^2 \;+\; \|W - U^\top\|_F^2} \;=\; \frac{2\,\langle W,\, U^\top\rangle_F}{\|W\|_F^2 + \|U\|_F^2} \;\in\; [-1, 1]. \tag{19}$$

The first form compares the energy of the pair's symmetric component against its antisymmetric component; the second, equivalent form is a normalized correlation between W and $U^\top$. The score is $+1$ exactly when $U = W^\top$ (tied feedback, as in equilibrium propagation), -1 exactly when $U = -W^\top$ (skew-symmetric), and 0 when W and $U^\top$ are orthogonal — e.g. for random, zero, or uncorrelated feedback. For convolutional blocks, $U^\top$ denotes the kernel of the adjoint operator (spatial flip and swap of the input/output channel axes), so $S = +1$ means the feedback convolution is the exact transpose of the forward convolution as a linear map. We report S per reciprocal pair at every epoch; under learned asymmetric feedback it tracks how much weight symmetry training discovers on its own.

Hyperparameters. Classification hyperparameters (Table 6) are shared across *all* algorithm variants compared — RBP, dyadic RBP, split dyadic RBP, feedforward and asymmetric connectivity, and the parameter-matched control — with only the algorithm/connectivity choice varying. Denoising hyperparameters are given in Table 7 for both optimizer arms.

Table 6: Classification hyperparameters (CIFAR-10 CNN, shared across all algorithm variants).

Hyperparameter	Value
Epochs	100
Batch size	32
Optimizer	SGD, momentum 0.9
Learning rate	$0.03 \to 2\times10^{-6}$ (cosine, no warmup)
Weight decay	5×10^{-5} (all parameters)
Gradient clipping	none
Nudging β	1.0 (split dyadic RBP: as listed in Table 1)
Dyadic α	$1/2$ (1 in the last row of Table 1)
Relaxation step size η	0.35 (all layers, both phases)
Tolerance (free / nudged)	2×10^{-3} / 2×10^{-3}
Iteration cap (free / nudged)	1560 / 1560
Damping factor	0.85
Traversal	forward sweep (free) / backward sweep (nudged)
Initialization	EBL uniform (see above)

Optimization details. Weight decay for SGD is coupled L_2 regularization (added to the gradient before the momentum accumulator, hence scaled by the schedule); for Adam it is applied decoupled (AdamW-style). Decay applies to all parameters, including biases and feedback kernels. Where gradient clipping is used (denoising), each parameter tensor is clipped independently to the stated norm. Non-finite gradient entries are zeroed. Learning rates follow cosine annealing without restarts (Loshchilov & Hutter, 2017) from the initial to the final value over the full run.

Table 7: Denoising hyperparameters (feedback denoiser, both datasets, encoder-only and U-Net variants, all connectivities).

Hyperparameter	Value
Epochs	100
Batch size	64
Optimizer	SGD, momentum 0.9
Learning rate	$10^{-2} \to 10^{-5}$ (cosine)
Weight decay	3×10^{-3}
Gradient clipping	1 (per parameter)
Nudging β	0.1
Relaxation step size η	per connectivity: ff/sym 1.0, skew 0.3, asym 0.5
Tolerance (free / nudged)	10^{-3} / 10^{-3}
Iteration cap (free / nudged)	1260 / 1200
Damping factor	0.95; nudged phase inherits $0.95\times$ free-phase η
Noise factor γ	1.0
Initialization	GOE, $V = 10^{-5}$
Seeds	3

Evaluation protocol. Inference is the free-phase relaxation alone (from zero states, same tolerance, caps, and damping as in training). For classification, the model with the highest per-epoch evaluation accuracy is retained and its test accuracy reported; as no validation split was reserved (val-fraction 0), the per-epoch evaluation set is the test set itself. For denoising, the best model is selected by lowest validation loss on the held-out 2,500-image split and evaluated once on the test set; validation/test noise is drawn freshly at each evaluation pass.

Other details. Code is implemented in JAX (Bradbury et al., 2018) (v0.6.2, Flax 0.10.7, Optax 0.2.5) with deterministic XLA/cuDNN kernels enabled; a single seed fully determines parameter initialization, data ordering, augmentation, and noise draws. Denoising simulations were run on a SLURM cluster (1 GPU, 32 GB host memory, 12 h wall-time per job; CUDA 12.2, cuDNN 9).

E Additional Gradient Comparison Results

The gradient curves in the main paper are for a randomly initialized network. Figure 4 shows the corresponding curves for dyadic RBP for a trained network.

Figure 5 shows the gradient alignment relative to BPTT for dyadic RBP and split dyadic RBP for $\alpha \in \{0, \frac{1}{2}, 1\}$. In all cases using a very small β is detrimental in Float32 precision. This is an artifact of the limited numerical precision (you throw away information when you downscale a small error signal several orders of magnitude) and vanishes in Float64. For $\alpha = 0$ and $\alpha = 1$ we see near perfect alignment to relative BPTT for both dyadic RBP and split dyadic RBP. For $\alpha = 1/2$ large β is detrimental for split dyadic RBP.

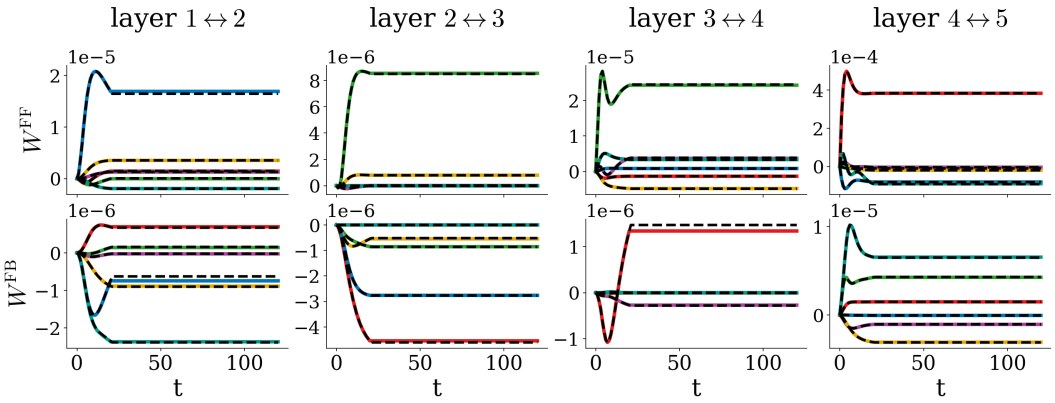
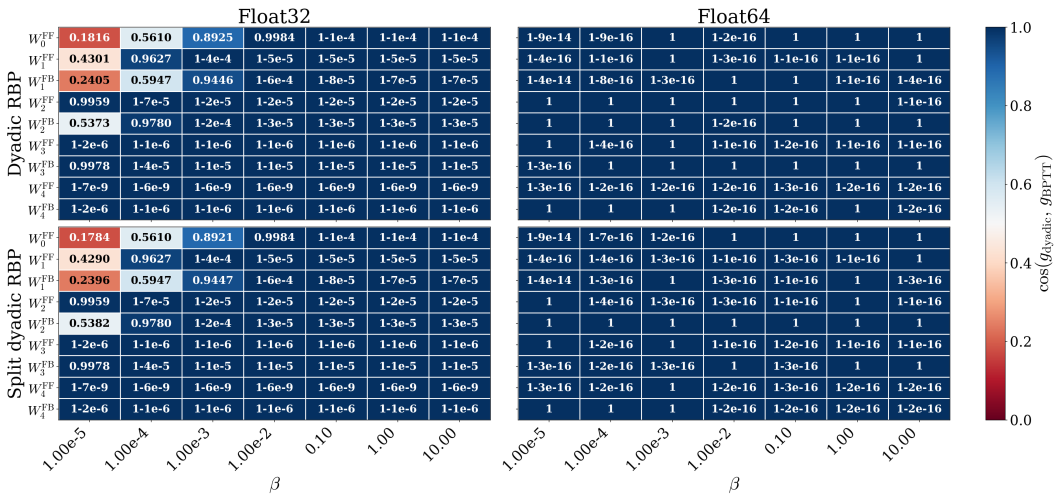


Figure 4: Supervised-phase gradients of adjoint RBP vs. dyadic RBP for the fully trained CNN of Section 4.1.

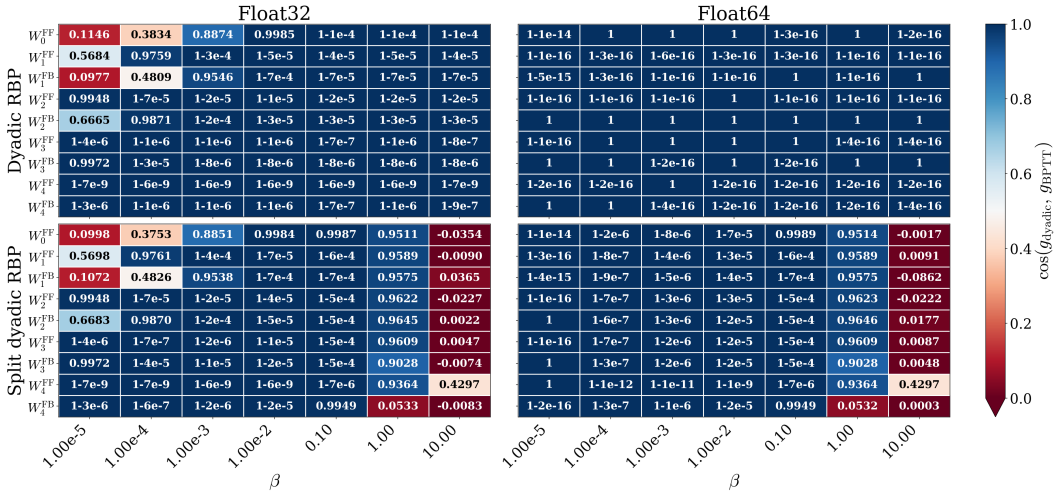
F Additional Classification Results

Table 8: Split dyadic performance on asymmetric CNNs trained on CIFAR-10 .These are additional results from 1 with alpha=1 under varying degrees of nudging ($\beta \geq 1$). Test accuracy (%) of split dyadic ($\alpha = 1$), mean $\pm$ std over seeds.

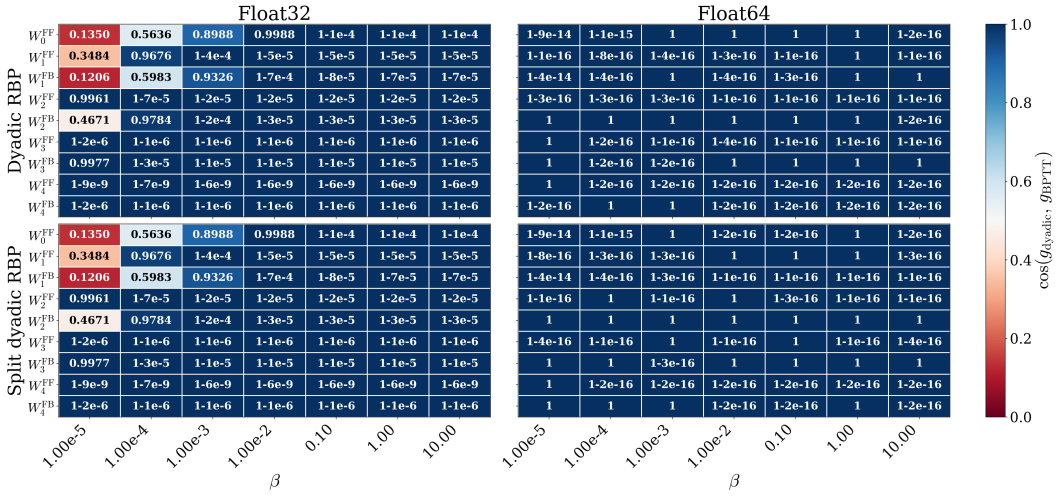
Nudging β	1	5	10
Accuracy (%)	92.75 ± 0.19	92.67 ± 0.13	92.83 ± 0.10



(a) $\alpha = 0$



(b) $\alpha = 1/2$



(c) $\alpha = 1$

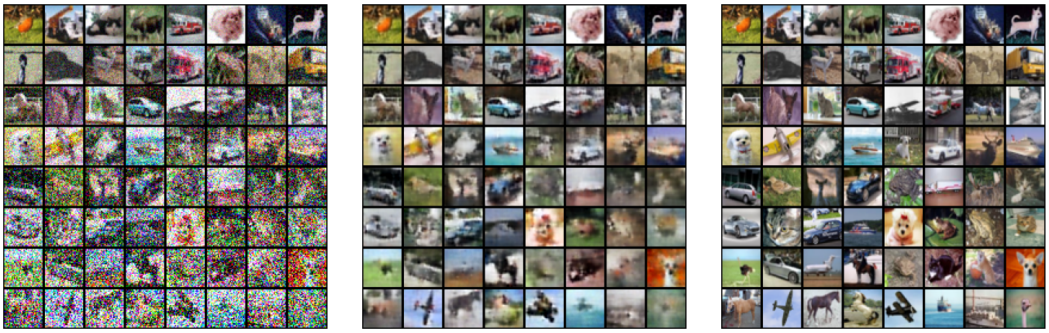
Figure 5: Cosine similarity of gradient estimates of dyadic RBP and split dyadic RBP in an asymmetric 5 layer CNN relative to BPTT with $\alpha = 0$, $\alpha = 1/2$ and $\alpha = 1$. To make the plots comparable the colorbar in **b** is clamped at 0.0 even though some cells are slightly negative.

G Additional Denoising Results

Below we present denoising results deferred from Section 4.2 Per-cell raw and normalized scores for every configuration are given in Table 9.

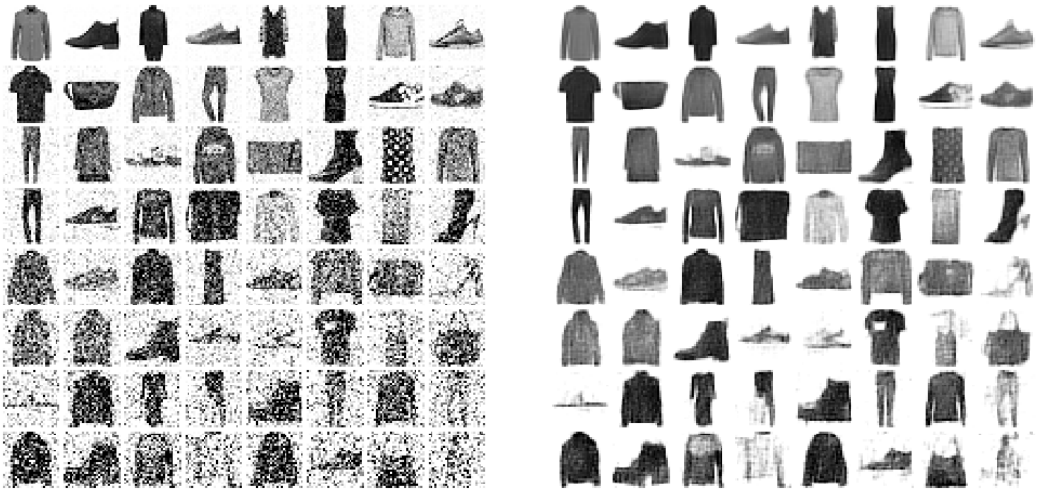
Table 9: Per-cell denoising results (95% train split, sgd recipe). **Raw MSE** is the summed per-image reconstruction error $\sum \frac{1}{2}(\hat{x}-x)^2$ on the test set, mean $\pm$ std over seeds. †: one seed failed (excluded from the statistics); *fail*: all seeds failed

Recon	Dataset	Conn.	Depth	Raw test MSE
recon@input	FMNIST	Asym	1	15.24 ± 0.07
		Asym	3	12.35 ± 0.17
		Asym	5	12.41 ± 0.23
		Sym	1	15.88 ± 0.13
		Sym	3	12.53 ± 0.08
		Sym	5	12.61 ± 0.09
		Skew	1	16.15 ± 0.04
		Skew	3	15.38 ± 0.05
		Skew	5	15.72 ± 0.62
recon@input	CIFAR10	Asym	1	38.23 ± 0.24
		Asym	3	32.54 ± 1.21
		Asym	7	30.81 ± 0.22
		Asym	8	30.1 ± 0.3
		Sym	1	41.85 ± 0.48
		Sym	3	37.34 ± 0.13
		Sym	7	38.12 ± 0.51
		Skew	1	41.01 ± 2.44
		Skew	3	37.13 ± 0.00
		Skew	7	37.55 ± 0.81
recon@output	FMNIST	Asym	1	14.09 ± 0.13
		Asym	3	10.36 ± 0.14
		Asym	5	10.32 ± 0.18
		Sym	1	16.36 ± 1.31
		Sym	3	24.51 ± 2.52
		Sym	5	43.69 ± 12.55
		Skew	1	15.12 ± 0.15
		Skew	3	13.40 ± 0.09
		Skew	5	16.03 ± 0.21
		FF	1	16.00 ± 0.03
		FF	3	10.77 ± 0.02
		FF	5	10.59 ± 0.08
recon@output	CIFAR10	Asym	1	35.83 ± 0.24
		Asym	3	32.36 ± 0.85
		Asym	7	31.28 ± 0.36
		Asym	9	28.8 ± 0.2
		Sym	1	40.92 ± 0.27
		Sym	3	45.95 ± 0.08
		Sym	7	fail
		Skew	1	37.26 ± 0.25
		Skew	3	$65.06^{\dagger}$
		Skew	7	fail
		FF	1	39.99 ± 0.32
		FF	3	30.41 ± 0.39
		FF	7	29.70 ± 0.37
		FF	9	29.90 ± 0.17



(a) Noisy input (b) Denoised (model) (c) Clean target

Figure 6: CIFAR-10 test-set denoising with the asymmetric denoiser trained on 9 layer U-Net via Dyadic RBP: noisy inputs **(a)**, model reconstructions **(b)**, and clean targets **(c)**.



(a) Noisy input (b) Denoised (model)

Figure 7: FASHION-MNIST test-set denoising: noisy inputs **(a)** and model reconstructions **(b)**. Produced with 3-layer asymmetric encoder-only denoiser, trained via Dyadic RBP

H ADAPTIVE STEP-SIZE DAMPING

Inference relaxes the state toward a root of the constraint $F(\theta, s) = s - G(f(s))$, with $G(f(s)) = Wf(s)+b$ the per-layer affine drive (W = the weight blocks θ of the main text; θ suppressed below), by per-layer damped iterations

$$s_\ell \;\leftarrow\; s_\ell - \eta_\ell\, F_\ell(s), \tag{20}$$

swept layer by layer. Each layer starts at a per-connectivity default $\bar{\eta}$ (feedforward/symmetric 1, skew-symmetric 0.3, asymmetric 0.5), and the heuristic multiplies a layer's step by $\rho = 0.95$ whenever its residual increases compared to previous cycle— a backtracking line search without rollback (Armijo, 1966). The alternative is to compute the step from local curvature. Both procedures are given in Figure 8.

Loss of contractiveness in asymmetric networks Near a fixed point s^* the dynamics are linear: the error $z = s - s^*$ evolves as $z \leftarrow (I - \eta J)\, z$ with $J = \partial F/\partial s$ (the transpose of the main text's $J(s)$; all quantities used here are transpose-invariant). Writing $\mu = \lambda_{\min}\big(\frac{1}{2}(J + J^\top)\big)$ and $\sigma = \|J\|_2$, the residual shrinks at every cycle whenever $\eta < 2\mu/\sigma^2$, and the iteration diverges beyond a somewhat larger threshold; rotational (antisymmetric) coupling produces spirals and, at the critical step, limit cycles (Figure 9). Good step sizes differ by connectivity. In a feedforward

Exact line search (`--optimal-eta`)

for cycle $t = 1, 2, \ldots$ **do**
  for layer ℓ in sweep order **do**
    $g \leftarrow F_\ell(s)$
    $h \leftarrow (\partial F_\ell / \partial s_\ell)\, g$ ▷ one JVP
    if $\langle g, h \rangle \le \varepsilon$ **then**
      $\eta \leftarrow \bar{\eta}_\ell$ ▷ fallback
    else
      $\eta \leftarrow \min\big(\langle g, g \rangle / \langle g, h \rangle,\ 2\bar{\eta}_\ell\big)$ ▷ cap
    end if
    $s_\ell \leftarrow s_\ell - \eta\, g$
  end for
end for

Cycle damping (default)

$\eta_\ell \leftarrow \bar{\eta}_\ell$, $r_\ell \leftarrow \infty$ for all ℓ
for cycle $t = 1, 2, \ldots$ **do**
  for layer ℓ in sweep order **do**
    $s_\ell \leftarrow s_\ell - \eta_\ell\, F_\ell(s)$
  end for
  for layer ℓ **do**
    $r'_\ell \leftarrow \|\Delta s_\ell\| / \|s_\ell\|$
    if $r'_\ell > r_\ell$ **then**
      $\eta_\ell \leftarrow \max(\rho\, \eta_\ell,\ \eta_{\min})$ ▷ damp
    end if
    $r_\ell \leftarrow r'_\ell$
  end for
  stop when $\max_\ell r'_\ell < \mathrm{tol}$
end for

Figure 8: The two step-size procedures: per-layer exact line search (left) and the damping heuristic (right).

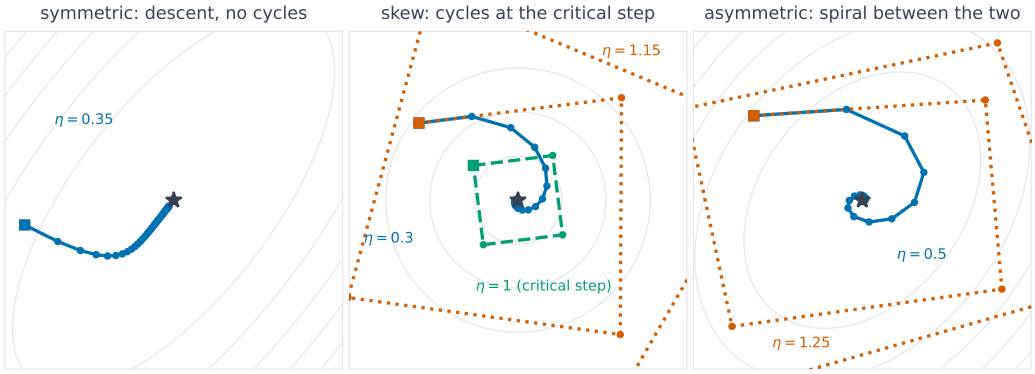


Figure 9: Toy relaxation $s_{t+1} = s_t - \eta(I - M)s_t$ (square: start; star: fixed point). Symmetric coupling descends the energy monotonically (left); skew coupling spirals in below the critical step, orbits exactly at it, and spirals out above (middle); asymmetric coupling interpolates (right).

network $\eta = 1$ solves $F = 0$ exactly in one forward sweep. In symmetric/Hopfield networks the $\eta = 1$ update exactly minimizes the Hopfield energy (Hopfield, 1984) one layer at a time, so convergence to some fixed point is guaranteed at every weight scale. While the skew-symmetric setting produces oscillations by definition, rotation can never destabilize the relaxation. Still, safe step shrinks with the coupling — about $2/(1 + \|KD\|^2) \approx 0.56$ at the measured coupling, which is why the default is 0.3. In the asymmetric setting μ and σ depend on the *learned* weights and the current activation pattern, so no fixed step is correct for every network and input. Instead we must estimate the correct step size dynamically.

The optimal eta On a locally linear piece the next residual after a step along F is exactly $F - \eta JF$, so the step that minimizes it has a closed form:

$$\eta^*(s) \;=\; \arg\min_{\eta} \; \|F(s - \eta F(s))\| \;=\; \frac{\langle F,\, JF \rangle}{\|JF\|^2}\,. \tag{21}$$

When $\langle F, JF \rangle \le 0$ no positive optimal step exists and the algorithm falls back to the default (we log the rate of occurrence of this, which in stable models, with our hyperparameters, is never). Computed *per layer* (Figure 8, left), the analogous quantity is identically 1 on our architectures, since no layer drives itself: smaller steps are owing only to *cross-layer* coupling.

Adaptive eta damping Figure 10 shows the step-size regimes on a toy asymmetric system: with sufficiently small step size, we contract monotonically, steps in the intermediate band converge with

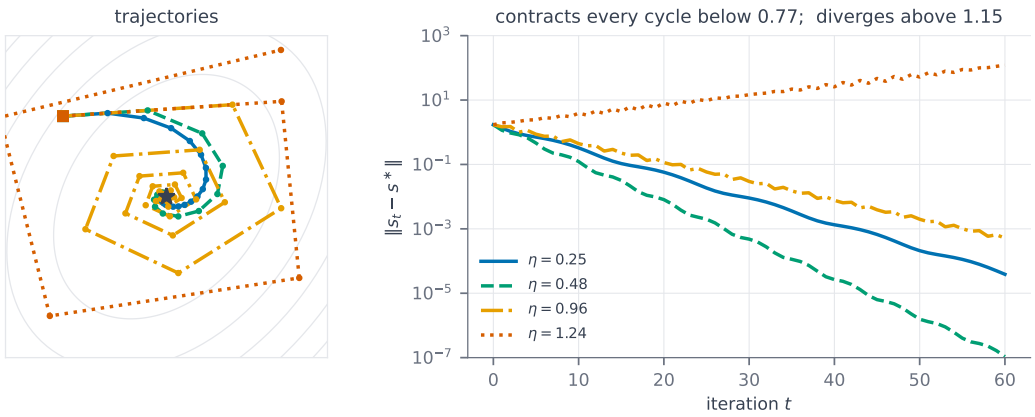


Figure 10: Various step sizes shown on an asymmetric system ($\mu = 0.51$, $\sigma = 1.42$). 0.77 is the largest step that contracts the residual at every iteration; the bound $2\mu/\sigma^2 = 0.50$ sits just under it. Between 0.77 and the divergence threshold 1.15 the we converge but with occasional residual growth, which triggers damping. Above 1.15 it diverges.

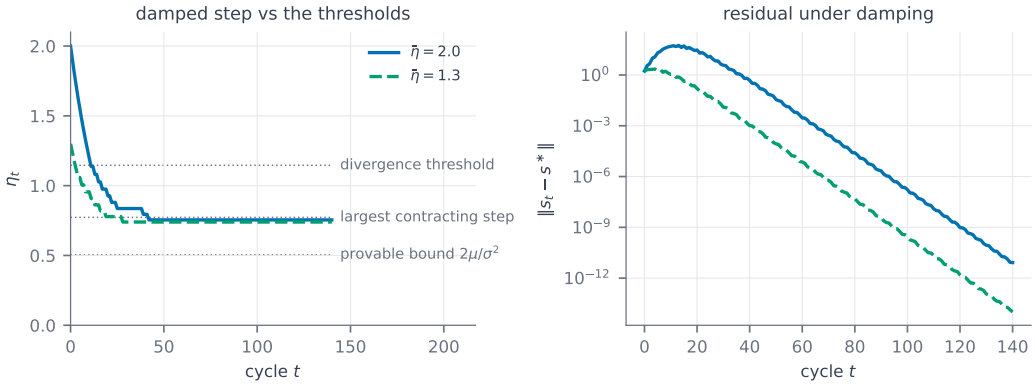


Figure 11: The damping rule finds the ceiling. From two bad initializations the step settles just below the largest contracting step, at 0.98 and 0.96 of it (left); the residual grows only during the brief backtracking transient, then contracts geometrically (right).

occasional residual growth, triggering damping. Larger steps diverge. Figure 11 runs the damping rule itself: started from deliberately bad steps ($\bar{\eta} = 2.0$ and 1.3, both non-convergent), it decreases until settling at 0.98 and 0.96 of the largest step that still contracts at every cycle — within 5% of the optimum using residual comparisons. In practice the same behavior appears across connectivities: damping is not triggered for feedforward and symmetric networks (where $\eta = 1$ is contractive and guarantees convergence), and settles the skew and asymmetric networks at stable step-size without tuning.